Platformski neovisna udaljena vizualizacija volumnih medicinskih podataka

Jozić, Krešimir

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Krešimir Jozić

PLATFORMSKI NEOVISNA UDALJENA VIZUALIZACIJA VOLUMNIH MEDICINSKIH PODATAKA

DOKTORSKI RAD

Zagreb, 2022.

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Krešimir Jozić

PLATFORMSKI NEOVISNA UDALJENA VIZUALIZACIJA VOLUMNIH MEDICINSKIH PODATAKA

DOKTORSKI RAD

Mentor: Prof. dr. sc. Željka Mihajlović

Zagreb, 2022.

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Krešimir Jozić

CROSS-PLATFORM REMOTE VISUALIZATION AND VOLUME RENDERING OF MEDICAL DATA

DOCTORAL THESIS

Supervisor: Professor Željka Mihajlović, PhD

Zagreb, 2022

Doktorski rad izrađen je na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva, na Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave.

Mentor: prof. dr. sc. Željka Mihajlović

Doktorski rad ima: 78 stranica

Doktorski rad br.: _____

O mentoru

Željka Mihajlović je rođena u Zagrebu 1965. godine.

Diplomirala je, magistrirala i doktorirala u računarstvu na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva (FER), Zagreb Hrvatska, redom 1988., 1993. i 1998.

Od veljače 1989. radi na Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva (FER). U studenom 2011. godine postaje izvanredna profesorica, a u srpnju 2013. godine redovita profesorica.

Vodila je međunarodni projekt „Modeling and Visualization of Complex Objects System", s industrijom, AVL-AST d.o.o. za Advanced Simulation Technologies Development, sa sjedištem u Zagrebu i AVL Graz - Instrumentation and Test Systems. Bila je voditeljica projekata „Animation of Articulated Kinematic Structures" i „Visualization of Multidimensional Data Sets" financirano od Ministarstva znanosti, obrazovanja i sporta, Republike Hrvatske.

Sudjelovala je u osam znanstvenih i tehnoloških projekata financiranih od strane Ministarstva znanosti, obrazovanja i sporta, Republike Hrvatske, uključujući: „Computing Environments for Ubiquitous Distributed Systems", "Adaptive scenarios control in the VR PTSD therapy" i "Optimization of renewable electricity generation systems connected in a microgrid". Objavila je oko pedeset radova u časopisima i u zbornicima konferencija u računalnoj grafici, računalnoj animaciji, rekonstrukciji i vizualizaciji.

Profesorica Mihajlović, član je IEEE, MIPRO, KoREMA te je član povjerenstva AMAC-FER (Almae Matris Alumni Croaticae).

About the Supervisor

Željka Mihajlović was born in Zagreb in 1965.

She received her bachelor's, master's, and doctoral degrees in computer science from the University of Zagreb, Faculty of Electrical Engineering and Computing (FER), Zagreb, Croatia, in 1988, 1993, and 1998, respectively.

Since February 1989, he has been working at the Department of Electronics, Microelectronics, Computer and Intelligent Systems at the University of Zagreb, Faculty of Electrical Engineering and Computing (FER). In November 2011, she became an associate professor, and in July 2013, a full professor.

She led the research project: "Modeling and Visualization of Complex Objects System", an international project with industry, AVL-AST d.o.o. for Advanced Simulation Technologies Development, based in Zagreb and AVL Graz - Instrumentation and Test Systems. She was the leader of the projects "Animation of Articulated Kinematic Structures" and "Visualization of Multidimensional Data Sets" funded by the Ministry of Science, Education, and Sports, Republic of Croatia.

She has participated in eight scientific and technological projects funded by the Ministry of Science, Education, and Sports, the Republic of Croatia, including: "Computing Environments for Ubiquitous Distributed Systems", "Adaptive scenarios control in VR PTSD therapy" and "Optimization of renewable electricity generation systems" connected in a microgrid ". She has published about fifty papers in journals and conference proceedings in computer graphics, computer animation, reconstruction, and visualization.

Professor Mihajlović is a member of IEEE, MIPRO, KoREMA, and a member of the AMAC-FER committee (Almae Matris Alumni Croaticae).

Sažetak

Radiološke snimke* pohranjene u DICOM obliku zbog svoje veličine mogu predstavljati problem korisnicima jer su računala na kojima se pregledavaju pod-optimalna. Učitavanje takvih snimki, kao i generiranih volumnih podataka, traje dugo što demotivira korisnike.

U slučaju da se snimke ne pregledavaju u ustanovi u kojoj su nastale, prenose se na optičkim ili USB medijima. Kako se snimke obično pregledavaju samo jednom, optički mediji se odbacuju, a to kao posljedicu ima onečišćenje okoliša.

Ako se snimke prenose preko Interneta i dalje ostaje problem pod-optimalnih računala ali i cijene usluge korištenja Interneta. Situacija je dodatno pogoršana COVID-19 pandemijom što otežava prijenos medija. Osim toga, rad od kuće je uzrokovao povećanu potražnju za poluvodičima, što je dovelo do nestašice i porasta cijene poluvodiča na tržištu.

Disertacija je usmjerena na rješavanje problema tako da predstavi rješenje za daljinsko iscrtavanje radioloških snimki. Rješenje je predstavljeno kroz tri dijela: izradu modela programske potpore, arhitekturu i tehničku analizu ostvarenog sustava. Provedeno istraživanje pokazuje da je moguće ostvariti prikazivanje pojedinačnih slojeva radiološke snimke gotovo trenutno što korisnicima olakšava rad te ubrzava postavljanje dijagnoze.

Ključne riječi: radiološke snimke, grafičko sklopovlje, iscrtavanje volumena, daljinsko iscrtavanje

*Koristi se izraz snimka umjesto slika radi toga što se slike mogu prikazati bez obrade (osim dekomprimiranja) dok kod DICOM zapisa to nije moguće.

Extended Abstract

CROSS-PLATFORM REMOTE VISUALIZATION AND VOLUME RENDERING OF MEDICAL DATA

Radiological images stored in DICOM format, due to their size, can be a problem for users because the computers on which they are viewed are sub-optimal. Loading such recordings, as well as generating volume data, takes a long time, which demotivates users.

In case the recordings are not reviewed in the institution where they were created, they are transferred on optical or USB media. As recordings are usually viewed only once, the optical media is discarded, resulting in environmental pollution.

If the recordings are transferred via the Internet, the problem of sub-optimal computers and the price of the Internet usage service remains. The situation is further aggravated by the COVID-19 pandemic, which makes media transmission difficult. In addition, working from home has caused an increased demand for semiconductors, leading to a shortage and an increase in the price of semiconductors in the market.

The dissertation is aimed at solving the problem by presenting a solution for the remote rendering of radiological images. The solution is presented in three parts: the creation of a program support model, architecture, and technical analysis of the realized system. The conducted research shows that it is possible to display the individual layers of the radiological image almost instantly, which makes it easier for users to work and speeds up the diagnosis.

Doctors and researchers in the field of medicine need to review various radiological images in their daily work. A commonly used notation for radiological images is DICOM. It is a standard published by the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA). In addition to the format of the record, the standard also defines the method of transferring records over the network and thus enables the integration of different systems such as scanners, servers, workstations, and PACS systems (Picture Archiving and Communication Systems). The standard consists of several chapters and also defines the method of access via REST web services (chapter 18 "Web Services").

Due to the complexity of the standard, there are a small number of software solutions that fully implement all its elements. Mostly specialized browsers are used that only display basic data and images. In addition, the standard allows the entry of a large amount of metadata, which makes reading difficult because some fields are not filled in or are filled with incomplete data.

Depending on the type of image, the records can be large from several tens of kB (dental images) to several GB (CT and MR scans). The display of DICOM images, in addition to specialized software solutions, in the case of large-sized images requires the use of workstations.

In everyday practice, DICOM images are recorded in one institution and viewed in another institution. Recordings are most often transferred on optical or USB media. The computers used to view the recordings are generally sub-optimal. Loading large images on such computers, if at all possible, is often time-consuming, which reduces the productivity and concentration of the doctors examining them.

The speed of reading radiological images can be of crucial importance for the patient. If the imaging is performed in one institution, and the diagnostics in another, it may take several days before a diagnosis is made. Ideally, the doctor reading the recording should have access to the recording without delay. This is only possible if the recording is available over the network. The conclusion is that the recording should be stored in the institution where it was recorded, and the doctors who read it, regardless of whether they are in the same institution or not, should have access via the network (LAN or Internet).

Medical recordings can be lost during transmission, and after reading they are archived and rarely used again. This means that the optical media that was used to transfer the recording is mostly used once, and after long-term storage, the media may be damaged and records may be lost. Although optical media are small in size, a large amount of discarded media still has an impact on environmental pollution. Accessing recordings over the network also solved the problem of a one-time use of optical media.

Accessing recordings over the network increases the complexity and amount of work for doctors. Instead of just opening a recording from optical or USB media, they now have to first find it, wait for the recording to be retrieved, and then open and review it. The speed of retrieving the recording depends on the characteristics of the network used, and in the case of using the Internet, the price of the service. There also remains the problem of sub-optimal computers that may not be able to open the image due to its large dimensions.

In addition to all the problems mentioned so far, the biggest problem is the COVID-19 pandemic, which further complicates the contact of doctors with patients, the transfer and handling of optical or USB media, but also the indirect consequences of the crisis, which the shortage of semiconductors on the market can be highlighted. In other words, if someone wants to invest in new computer equipment, he cannot get it because it is not available on the market, which, according to the law of supply and demand, leads to an increase in equipment prices and worsens the situation even more.

By analyzing all the observed problems, the conclusion is reached that the recordings should be stored where they were created, and only the layers selected by the user at the time of viewing should be transferred. The system should be designed so that users do not experience a delay in

transmission and that they do not need to invest in new equipment. It is also necessary to ensure that investments from the outside of the institution that stores the recordings are minimal or none at all, that is, that the existing circuitry is used in the best possible way.

To build a system that solves numerous problems, it is important to carefully analyze each problem, and when finding a solution, be careful not to create a new problem. For example, by processing recordings on the servers, the problem of sub-optimal user computers is transferred to the servers. In this case, the problem is reduced because the servers have more disk space, processors with more cores, and more memory.

The thesis consists of six chapters: Chapter 1 "Introduction", Chapter 2 "Related research", Chapter 3 "A model of efficient program support and procedures for multi-user remote visualization of volume medical data", Chapter 4 "Platform-independent software support architecture based on visualization-communication workflow", Chapter 5 "Qualitative and quantitative technical analysis and evaluation of the inherent performance of the built software system", Chapter 6 „Conclusion”, and three appendices.

In the first chapter, an introduction to the field of radiological image presentation is given, the motivation is presented, the approach and contribution are presented, and the organization of the work is presented.

In the second chapter, related research is listed. It can be seen that the problem of displaying radiological images is repeated cyclically. First, the circuitry becomes more complex and enables the user to display large images locally, and then with the development of scanners, the dimensions of the image increase, which makes local display impossible, and the cycle starts from the beginning.

In the third chapter, a model of effective program support and procedures for multi-user remote visualization of volume medical data is presented. The individual components of the system and the method of processing recordings, as well as the interaction between individual parts of the system, are presented in detail.

In the fourth chapter, a platform-independent architecture of program support based on visualization-communication workflow is presented. The technologies used in the creation of the software solution are presented.

In the fifth chapter, a qualitative and quantitative technical analysis and evaluation of the inherent performance of the built program system are given. Load times for recordings on the server are shown, as well as load times for individual layers which are around 100ms or less giving users a sense of instant responsiveness.

The last chapter presents the conclusions and contributions of this dissertation. Guidelines for future research on topics that were not covered by this dissertation are provided.

Appendix A contains an example of a communication interface specification written in

YAML form.

Appendix B shows an example of DICOM record metadata.

Appendix C consists of screenshots of the frontend.

From the test results, it can be concluded that the rendering time in cases of real network conditions is masked by the latency and bandwidth of the network. The duration of the transfer depends on the size of the image, and the size on the algorithm used to compress the image. The WebP algorithm gave the best results. Newer image compression algorithms, such as HEIF, AVIF and JPEG XL, were not tested because at the time of writing they are not supported on web browsers or there are no libraries for the Go programming language.

The main scientific contributions of this dissertation are:

• A model of effective program support and procedures for multi-user remote visualization of volume medical data was created

With the motivation to solve problems from practice (chapter 1), the research of the area was carried out (chapter 2) and the requirements were defined for the system. With the system requirements defined, the system model was first created and the communication between the system parts was defined (chapter 3). When creating a system model, it is necessary to pay attention to the recurring trend of the development of technologies and the increase in the size of records, i.e. to create a solution that does not need significant processing after a certain time. A significant part of the research is devoted to the security of data transmission over the network, as well as local data storage.

• A platform-independent software support architecture based on visualization-communication workflow was created

Based on the model and communication procedures described in chapter 3, the system architecture was constructed and the software solution was implemented. The system architecture is designed to be platform-independent. This means that it should be able to run on any modern processor architecture and under any operating system that supports OpenGL 4.5 or 4.6. The OpenGL 4.5 standard was published in 2014, which means that the circuitry used must not be older than eight years. Given that it is an unwritten rule that servers should be replaced every five years, the eight-year requirement allows for the extension of the life of the circuit and savings for users. The most valuable part of the system is the data recorded in databases and DICOM records. The method of data protection in databases is described, and for DICOM records, storage on RAID arrays or SAN/NAS devices is recommended. A software solution was created that allows the user to easily use the system without the need to know the system architecture. By combining different asynchronous programming procedures, the reduction of data transfer latency is ensured, so that the user has a feeling of immediate system response. The

software solution is made so that it can be copied and run without the need to install, and system maintenance is reduced to a minimum.

• A qualitative and quantitative technical analysis and evaluation of the inherent performance of the built program system was carried out

The records are selected so that the footprint is in the range of about 1 MB to 1 GB, and that each subsequent one is about an order of magnitude larger than the previous one.

The tests were performed by running a system with one web server, one peak server, one enterprise-level server, and one render node. Records were viewed on the same computer in three ways: without network simulation, with 4G network simulation, and with 3G network simulation.

The occupancy of the input 2D texture and the storage 3D texture is calculated. After displaying an individual image, the occupancy of an individual output 2D texture was calculated from the screen dimensions. An output texture containing only one color channel was used to draw layers and volumes. A single-pass program for shading image elements based on ray emission and a limit of 100 steps was used to render the volume. This limit was chosen to make the test conditions the same for all images, although in the practical application the number of steps would depend on the dimensions of the image.

Keywords: radiological images, graphic hardware, volume rendering, remote rendering

Sadržaj

Poglavlje 1

Uvod

1.1 Uvodna razmatranja

Liječnici i istraživači iz područja medicine u svakodnevnom radu imaju potrebu za pregledavanjem različitih radioloških snimki. Često korišteni zapis radioloških snimki je DICOM. To je norma objavljena od strane organizacija American College of Radiology (ACR) i National Electrical Manufacturers Association (NEMA) [1]. Norma osim formata zapisa definira i način prijenosa zapisa preko mreže te tako omogućava integraciju različitih sustava kao što su skeneri, poslužitelji, radne stanice i PACS sustavi (engl. *Picture Archiving and Communication Systems*). Norma se sastoji od većeg broja poglavlja [2], a definira i način pristupa preko REST web usluga (poglavlje 18 „Web Services").

Zbog složenosti norme postoji mali broj programskih rješenja koja u potpunosti implementiraju sve njene elemente [3]. Uglavnom se koriste specijalizirani preglednici koji samo prikazuju osnovne podatke i slike [4]. Osim toga, norma dozvoljava unos velike količine metapodataka što otežava očitavanje jer neka polja nisu popunjena ili su popunjena s nepotpunim podacima. Ovisno o vrsti snimke, zapisi mogu biti veliki od nekoliko desetaka kB (dentalne snimke) do nekoliko GB (CT i MR skenovi). Prikaz DICOM snimki, osim specijaliziranih programskih rješenja, u slučaju slika velikih dimenzija zahtijeva upotrebu radnih stanica.

U svakodnevnoj praksi DICOM snimke se snimaju u jednoj ustanovi, a gledaju u drugoj ustanovi. Snimke se najčešće prenose na optičkim ili USB medijima. Računala koja se upotrebljavaju za pregledavanje snimki uglavnom su pod-optimalna. Učitavanje velikih snimki na takvim računalima, ako je uopće moguće, često je dugotrajno što smanjuje produktivnost i koncentraciju liječnika koji ih pregledavaju [5].

1.2 Motivacija

Brzina očitanja radioloških snimki može biti od presudne važnosti za pacijenta. U slučaju da se snimanje obavlja u jednoj ustanovi, a dijagnostika u drugoj, može proći nekoliko dana prije nego se postavi dijagnoza. Idealno bi bilo da liječnik koji očitava snimku ima pristup snimci bez odgode. Ovo je moguće jedino ako je snimka dostupna preko mreže. Nameće se zaključak da snimka treba biti pohranjena u ustanovi u kojoj je snimljena, a liječnici koji je očitavaju, bez obzira da li su u istoj ustanovi ili ne, trebaju imati pristup preko mreže (LAN ili Internet).

Medicinske snimke prilikom prijenosa mogu se izgubiti, a nakon očitavanja se arhiviraju te ih se rijetko kada ponovno koristi. To znači da se optički medij koji je poslužio za prijenos snimke uglavnom jednokratno koristi, a nakon dugotrajne pohrane može doći do oštećenja medija i gubitka zapisa. Iako su optički mediji malih dimenzija, velika količina odbačenih medija ipak ima utjecaja na zagađenje okoliša. Pristupom snimkama preko mreže također je riješen problem jednokratne upotrebe optičkih medija.

Pristupom snimkama preko mreže liječnicima se povećava složenost i količina posla. Umjesto da samo otvore snimku s optičkog ili USB medija, sada je prvo trebaju pronaći, pričekati da se snimka dohvati te je otvoriti i pregledati. Brzina dohvaćanja snimke ovisi o karakteristikama mreže koja se koristi, a u slučaju korištenja Interneta i cijena usluge. Također ostaje problem pod-optimalnih računala koja možda ne mogu otvoriti sliku zbog velikih dimenzija.

Uz sve do sada navedene probleme, najveći problem je COVID-19 pandemija koja dodatno otežava kontakt liječnika s pacijentima, prijenos i rukovanje optičkim ili USB medijima, ali i indirektnim posljedicama krize od kojih se može istaknuti nestašica poluvodiča na tržištu. Drugim riječima, ako netko poželi investirati u novu računalnu opremu, ne može je nabaviti zbog toga što je nema na tržištu, a što po zakonu ponude i potražnje dovodi do porasta cijena opreme i još više pogoršava stanje.

Analizom svih uočenih problema, dolazi se do zaključka da snimke trebaju biti pohranjene gdje su i nastale, a prenositi treba samo slojeve koje odabere korisnik u trenutku pregledavanja. Sustav treba biti konstruiran tako korisnici ne osjete kašnjenje u prijenosu te da ne trebaju investirati u novu opremu. Također je potrebno osigurati da investicije sa strane ustanove koja pohranjuje snimke budu minimalne ili nikakve, odnosno da se postojeće sklopovlje iskoristi na što bolji način.

1.3 Pristup i doprinos

Za izgradnju sustava koji rješava brojne probleme bitno je pažljivo analizirati svaki pojedini problem, a prilikom pronalaska rješenja paziti da se ne stvori novi problem. Npr. obradom snimki na poslužiteljima problem pod-optimalnih računala korisnika prebacuje se na poslužitelje.

U tom slučaju problem je smanjen jer poslužitelji imaju veći diskovni prostor, procesore s više jezgri te više radne memorije. Jedan od doprinosa ovog rada je izrada modela sustava koji omogućuje daljinski prikaz medicinskih snimki. Drugi doprinos je implementacija programskog rješenja koje se sastoji od 4 sloja, a omogućava korisnicima odabir pacijenata, pronalazak snimki, pregledavanje snimki, te unos i pohranu dijagnoze. Posljednji doprinos je provedena tehnička analiza i vrednovanje svojstvenih performansi sustava koji su pokazali da je vrijeme učitavanja pojedinog sloja snimke gotovo trenutan.

1.4 Organizacija rada

U ovom poglavlju dan je uvod u područje prikaza radioloških snimki, iznesena je motivacija, prikazani su pristup i doprinos te je izložena organizacija rada.

U drugom poglavlju navedena su srodna istraživanja. Vidljivo je da se problem prikaza radioloških snimki ciklički ponavlja. Prvo sklopovlje postane složenije i omogući korisniku lokalno prikaz velikih snimki, a onda se razvojem skenera povećaju dimenzije slike što onemogući lokalni prikaz te ciklus krene iz početka.

U trećem poglavlju prikazan je model učinkovite programske potpore i procedure za višekorisničku udaljenu vizualizaciju volumnih medicinskih podataka. Detaljno su prikazane pojedine komponente sustava i načina obrade snimki, kao i interakcija između pojedinih dijelova sustava.

U četvrtom poglavlju prikazana je platformski neovisna arhitektura programske potpore zasnovana na vizualizacijsko-komunikacijskom radnom toku. Prikazane su tehnologije korištene u izradi programskog rješenja.

U petom poglavlju dana je kvalitativna i kvantitativna tehnička analiza i vrednovanje svojstvenih performansi izgrađenog programskog sustava. Prikazana su vremena učitavanja snimki na poslužitelju, kao i vremena učitavanja pojedinih slojeva koja su oko 100 ms ili manje što daje korisnicima osjećaj trenutnog odziva [5].

U posljednjem poglavlju izloženi su zaključci i doprinosi ove disertacije. Navedene su smjernice za buduća istraživanja na temama koje nisu bile obuhvaćene ovom disertacijom.

Dodatak A sadrži primjer specifikacije komunikacijskog sučelja zapisanom u YAML obliku.

Dodatak B prikazuje primjer metapodataka o DICOM zapisu.

Dodatak C sastoji se od snimki zaslona web stranice.

Poglavlje 2

Srodna istraživanja

Primjena računala za vizualizaciju u medicini počinje sredinom 1970-ih godina prošlog stoljeća. Tada su se pojavili prvi primjeri primjene 3D vizualizacije u računalnoj tomografiji. Danas se za dijagnostiku koriste brojni modaliteti (CT, MR, PET, US itd.). Ostale bitne primjene su edukacija, vježbanje operativnih procedura, pred-operativno planiranje i telemedicina [6].

U početku su se koristile radne stanice sa specijaliziranim programskim rješenjima. Taj period je trajao dosta dugo jer su osobna računala bila pod-optimalna za vizualizaciju 3D podataka. Tek početkom 2000-ih godina grafičko sklopovlje na osobnim računalima dobiva podršku za 3D teksture što je osnovni preduvjet za kvalitetnu vizualizaciju volumena u stvarnom vremenu. Sredinom 2000-ih godina pojavljuje se programirljivo grafičko sklopovlje. Prvi specijalizirani programski jezik za programiranje grafičkog sklopovlja je Cg tvrtke NVIDIA. Ubrzo nakon toga pojavljuju se i platforme za paralelnu obradu na grafičkom sklopovlju. Prva takva platforma bila je CUDA, također od tvrtke NVIDIA. Dolazi do razvoja velikog broja algoritama za vizualizaciju volumena. Javljaju se i primjeri predobrade podataka, npr. konvolucijsko filtriranje ili FFT transformacija. Grafički procesori sastoje se od stotina ili tisuća vektorskih procesorskih jedinica pa obrada velikih snimki koja zahtijeva nekoliko stotina ili tisuća milijardi operacija traje kratko.

U međuvremenu je došlo do daljnjeg razvoja uređaja za prikupljanje snimki (skenera) te do porasta dimenzija snimki. Npr. za učitavanje snimke dimenzija 2048x2048x2048 koja pojedinu točku pohranjuje u podatku tipa float potrebno je 34 GB radne memorije. Za zahtjevnije obrade kao što je 4D obrada snimke dimenzija 512x512x446x20 korištenjem 14 filterskih elemenata tipa float memorijski zahtjevi rastu na 131 GB. Moderno grafičko sklopovlje na osobnim računalima raspolaže s do 8 GB radne memorije pa je za implementaciju algoritma kojemu je potrebno 131 GB radne memorije, potrebno je segmentirati regije na dimenzije koje stanu u memorijski ograničen prostor. To znači da je učestao prijenos preko sabirnice između centralnog procesora i grafičkog procesora što dovodi do pauza u izračunima, a time se poništava visoko-paralelna obrada na grafičkom procesoru [7].

2.1 Vizualizacija volumena

Za vizualizaciju radioloških snimki postoje tri grupe algoritama [8]:

•Višeravninska reformacija (engl. *multiplanar reformation*)

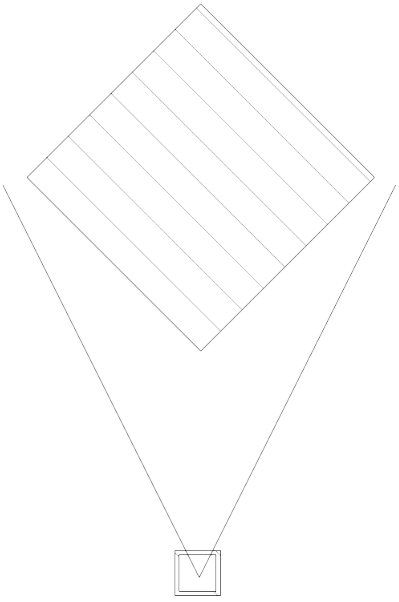
•Iscrtavanje površine (engl. *surface rendering*)

•Direktno iscrtavanje volumena (engl. *direct volume rendering*)

Najčešće korištena grupa algoritama je direktno iscrtavanje volumena te je ona detaljno istražena u ovom radu.

Prvi algoritmi za direktno iscrtavanje volumena temelje se na 2D projekcijama. Jedan takav algoritam je projekcija maksimalnog intenziteta (engl. *Maximum Intensity Projection*) kod kojega se u ravnini vizualizacije projiciraju točke najvećeg intenziteta koji padaju na putu paralelnih zraka koje se prate od gledišta do ravnine projekcije.

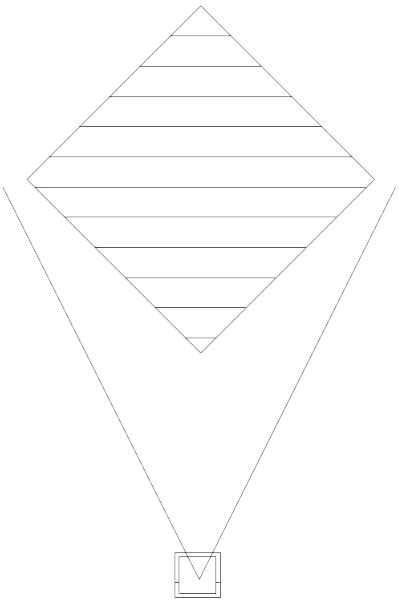
Nakon toga nastaju algoritmi temeljeni na teksturama. Moguća su dva pristupa: odsječci poravnati s objektom ili odsječci poravnati s pogledom. Drugi pristup daje bolju kvalitetu slike ali ga je u vrijeme nastanka bilo moguće izvesti samo na skupim radnim stanicama. Algoritam je moguće optimirati korištenjem različitih razina detalja nižih rezolucija pa se postižu rezultati uštede memorije do 43% s neznatnim gubitkom kvalitete ili s uštedom od 71% s vidljivim gubitkom kvalitete [9].



Slika 2.1 – Odsječci poravnati s objektom

Nakon toga nastaju algoritmi temeljeni na praćenju zrake (engl. *raytracing*). Ovi algoritmi daju najbolju kvalitetu slike ali se zbog velike količine računskih operacija sporo izvode. Ovom grupom algoritama moguće je vizualizirati objekte koje nije moguće vizualizirati ostalim grupama algoritama, a to su objekti koju su prozirni, reflektirajući i nepravilnog oblika [10].

Najčešće korišteni algoritmi za direktno iscrtavanje volumena su algoritmi temelji na odašil-



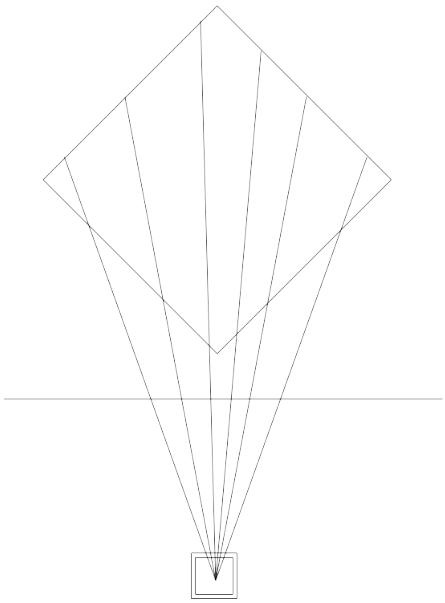
Slika 2.2 – Odsječci poravnati s pogledom

janju zrake (*eng. raycasting*). Algoritmi su jednostavni, a daju vrlo dobre rezultate. Iz položaja kamere odašilju se zrake kroz svaki element slike koja se nalazi između kamere i objekta. Zraka prolaskom kroz objekt akumulira vrijednost svjetline. Uvjeti zaustavljanja algoritma su akumulacija do maksimalnog intenziteta koji se može prikazati ili do izlaska zrake iz objekta. Algoritme je moguće optimirati tako da se preskače prazan prostor te na taj način smanjuje broj računskih operacija. Na grafičkom sklopovlju novijeg datuma moguće je iscrtavati više objekata istovremeno [11]. Klasični algoritmi uzorkuju elemente volumena na jednakim udaljenostima duž zrake. Prilagodbom načina uzorkovanja moguće je smanjiti broj uzorkovanja za 50% do 80% bez značajnog gubitka kvalitete [12]. Najnovija istraživanja pokazuju prilagodbu algoritma za iscrtavanje nekonveksnih objekata [13]. Razvijeni su radni okviri koji korisnicima olakšavaju implementaciju rješenja za interakciju s volumnim podacima [14] [15] [16].

2.2 Korištenje mobilnih uređaja

Široka primjena računalne grafike na mobilnim uređajima (mobiteli, tableti) počinje prije desetak godina kada se pojavljuju uređaji s velikim zaslonima i specijaliziranim grafičkim sklopovljem. U nedostatku sklopovske podrške za 3D teksture pojavljuju se algoritmi koji nizove tekstura slažu u jednu veliku teksturu kao pločice. Izbjegavaju se složeni algoritmi radi produljenja trajanja baterije [17].

Razvojem grafičkog sklopovlja, mobilni uređaju dobivaju podršku za 3D teksture. Međutim, 3D teksture su ograničenih dimenzija relativno malih u odnosu na one koje su podržane od strane grafičkog sklopovlja na računalima. Razvijaju se algoritmi koji jedan veliki volumen razbijaju na nekoliko manjih, npr. jedna kocka razbija se na 8 manjih [18].



Slika 2.3 – Odašiljanje zraka

Daljnjim razvojem grafičkog sklopovlja postaje moguće iscrtavati volumen većih dimenzija, uz izmjenu većeg broja slika u sekundi. Rezultati testiranja na DICOM slici rezolucije 512x512x144 korištenjem iscrtavanja volumena temeljenog na teksturama daje rezultat od 48.5 slika u sekundi. Korištenje iste dimenzije teksture i algoritma za direktno iscrtavanje volumena daje rezultat od 23.8 slika u sekundi [19].

2.3 Udaljeno iscrtavanje

Razvoj udaljenog iscrtavanja radioloških snimki potaknut je potrebom malih bolnica ili bolnica u ruralnim područjima [20]. Takve bolnice ne raspolažu grafičkim radnim stanicama, a trebaju pregledavati radiološke snimke. Predloženo je rješenje koje dohvaća DICOM slike pohranjene lokalno na nekom mediju ili dohvaćene direktno sa skenera te ih preko Java appleta unutar web preglednika na klijentskom računalu šalje na udaljenu radnu stanicu. Ovakva rješenja najčešća su u periodu 2001. - 2013. g. jer od 2013. g. web preglednici počinju ukidati podršku za Java applete sve do 2017. g. kada je ta tehnologija u potpunosti napuštena. Ne razmatra se brzina iscrtavanja niti vrijeme transporta preko mreže nego ukupna brzina dijagnostike koja je u svakom slučaju neusporedivo brža nego da se snimke prenose na fizičkom mediju te očitavaju na udaljenoj lokaciji.

Radni okvir SparkMed [21] koristi računarstvo u oblaku tako da stvara samoorganizirajuću mrežu poslužitelja s DICOM snimkama na koje se korisnici spajaju preko 4G mreže pomoću prijenosnih uređaja (prijenosna računala, mobiteli). Iscrtavanje se radi udaljeno, a konačna slika prenosi se i prikazuje na korisnikovom uređaju. Postignuto je vrijeme odziva od 5 sekundi od zahtjeva za slikom do prikaza na zaslonu korisnikova uređaja.

Korisnici umjesto korištenja web usluga definiranih DICOM normom često implementiraju rješenja koja odgovaraju njihovoj infrastrukturi. Starija rješenja su troslojna (web preglednik, web poslužitelj, čvor za iscrtavanje) te koriste VRML ili X3D kao formate za prijenos i pohranu podataka [22]. Novija rješenja su četveroslojna (web preglednik, web poslužitelj, poslužitelj DICOM datoteka, čvor za iscrtavanje) te koriste HTML5 i WebGL na strani klijentskog računala [23] dok na strani poslužitelja korištene tehnologije variraju. Usporedbom starijih (Java, Flash) i novijih tehnologija (HTML5) utvrđeno je da su Java i HTML5 bliski po performansama, ali HTML5 ne treba nikakve dodatke ni instalacije programa, već je sve potrebno definirano normom i implementirano unutar web preglednika [24].

Istraživanja interaktivnog udaljenog iscrtavanja u stvarnom vremenu idu u smjeru računalnih igara [25]. U slučaju kada se želi igrati računalna igra, a ne postoji mogućnost instalacije ili je lokalno računalo pod-optimalno (npr. u hotelima), moguće je izvođenje računalne igre na udaljenom računalu. Isprobana su dva pristupa. U slučaju da se koriste slabija računala sa zaslonima niže rezolucije, od korisnika se primaju naredbe, prenose na poslužitelj i tamo obrađuju. Rezultat je niz gotovih slika koje se vraćaju korisniku kao video. Taj video mora biti sažet jer inače ne prenosi dovoljan broj slika u sekundi i korisnik nema dobar osjećaj interakcije sa sustavom. U drugom slučaju, kada se koriste računala boljih performansi i sa zaslonima više rezolucije, postupak je sličan. Korisniku se ne vraćaju gotove slike nego zapakirani nizovi naredbi grafičkom sklopovlju koje lokalno obavlja iscrtavanje te prikazuje sliku korisniku.

2.4 Sažimanje

Daljinski iscrtane slike prenose se preko Interneta do krajnjeg korisnika. Količina podataka koju je potrebno prenijeti ovisi o rezoluciji slike. Nekomprimirana slika koja prikazuje nijanse sive boje zauzima 8 bita to točki. IBRAC [26] način sažimanja koristi svojstva volumnog iscrtavanja tako da uklanja redundanciju u podacima unutar jednog okvira slike i između više okvira slike te tako postiže sažimanje zapisa do 0.1 bit po točki.

Neka rješenja ne razmatraju omjer nego brzinu sažimanja. Ako se preko veze ograničene brzine želi prenijeti što više slika u jedinici vremena, osim veličine slike bitno je koliko traje obrada na poslužitelju. Poslužitelj prvo mora iscrtati volumen, a nakon toga rezultirajuću sliku sažeti te je poslati korisniku. Korištenjem postojećih algoritama za sažimanje slika koji se koriste na centralnom procesoru, vrijeme sažimanja može biti veće nego vrijeme iscrtavanja na grafičkom sklopovlju. Invire sustav [27] nakon iscrtavanja korištenjem OpenGL-a, obavlja sažimanje korištenjem CUDA platforme. Rezultati pokazuju da se slike rezolucije 1024x1024 ili veće isplati sažeti jer vrijeme paralelnog sažimanja pa slanja traje kraće od slanja nekomprimirane slike.

2.5 Zaštita podataka

Vrijedni 3D modeli kao što su skenovi kulturne baštine mogu zahtijevati zaštitu od neovlaštenog korištenja, ali istovremeno prikaz i interakcija sa sadržajem treba biti dostupna široj publici [28].

Mogući oblici napada su:

•Reverzni inženjering 3D modela

•Izmjene u programskom rješenju za pregledavanje slika

•Izmjene u upravlja čkom programu grafičkog sklopovlja

•Rekonstrukcija iz spremnika okvira (engl. *framebuffer*)

•Rekonstrukcija iz generirane slike

Postupci zaštite su:

•Iscrtavanje samo korištenjem programa, bez pomo ći grafičkog sklopovlja

•Hibridno programsko/sklopovsko iscrtavanje

•Deformacija geometrije

•Sklopovsko dekriptiranje na grafi čkom procesoru

•Iscrtavanje temeljeno na slikama

•Udaljeno iscrtavanje

Kao rješenje pokazalo se da nije dovoljno koristiti samo jednu strategiju obrane nego više njih u kombinaciji. Najbolje je koristiti udaljeno iscrtavanje jer je potrebno obraniti samo jedan poslužitelj, a sa strane poslužitelja moguće je u konačnu sliku umetnuti različite oznake ili deformacije.

2.6 Poboljšanja kvalitete slike

Volumno iscrtavanje može dati lošu kvalitetu slike ako su ulazni podaci niske rezolucije. Gube se detalji raznih organa, kostiju, žila itd. Pokazuje se da je moguće poboljšati kvalitetu rezultirajuće slike primjenom postupka preslikavanja neravnina (engl. *bump mapping*). Ovaj postupak je vrlo čest u računalnoj grafici, a izvodi se tako da se normale željenih neravnina pohrane u vektorskom obliku kao elementi tekstura te se prilikom izračuna osvjetljenja kombiniraju s objektom za koji se računa osvjetljenje i tako se dobiva efekt da površina glatkog objekta izgleda kao da je neravna. Pokazuje se da utjecaj postupka preslikavanja neravnina ima utjecaj od 1% na trajanje iscrtavanja volumena [29].

Kod klasičnih algoritama prijenosna funkcija koja preslikava fizička svojstva objekta u optička svojstva obično je fiksna ili se zapisuje u 1D teksturu te pretražuje po njoj na osnovu predefiniranog kriterija. Razvijeno je rješenje koje dinamički izračunava prijenosnu funkciju na osnovu Kubelka-Munk teorije o širenju svjetla u paralelnim obojenim slojevima zamućenog medija. Rješenje daje značajno poboljšanje slike, ali isto tako značajno usporava iscrtavanje i

nije ga moguće koristiti za primjene iscrtavanja u stvarnom vremenu [30].

2.7 Animacija

U nedostatku grafičkog sklopovlja i specijaliziranih preglednika, korištenjem MATLAB-a napravljeno je rješenje za pretvorbu višeslojnih DICOM snimki u filmski zapis [31]. Kako se višeslojni zapisi obično pregledavaju sloj po sloj, pretvorbom u filmski zapis način rada sa snimkama ostaje isti. Provedenim testiranjima na AVI, MOV i MP4 zapisima ustanovljeno je da dolazi do nezanemarivog gubitka kvalitete slike. To se može i očekivati jer su navedeni zapisi napravljeni tako da što više sažmu slike, a gubitak kvalitete se prikriva brzom izmjenom slika. Kod radioloških zapisa ne smije doći do gubitka kvalitete jer se mogu previdjeti sitni detalji koji su možda bitni.

2.8 Kombinirano lokalno-udaljeno iscrtavanje

Općenito rješenje za udaljeno iscrtavanje koristi postojeće programe koji prikazuju slike kod udaljenih klijenata tako da se ne rade nikakve preinake na postojećim programima [32]. Ostvarena su dva rješenja. Prvo rješenje koristi GLX protokol*. X Window System je klijent/poslužitelj arhitektura samo što se u većini slučajeva koristi na istom računalu. Rješenje koristi tu arhitekturu tako da se naredbe zadaju na klijentskom računalu, prenose na poslužtelj koji radi iscrtavanje, te rezultati vraćaju klijentu koji ih onda prikazuje. Drugo rješenje je da se iscrtavanje radi na poslužitelju, a naredbe i rezultati između klijenta i poslužitelja prenose preko VNC protokola. Prvi način ima ograničenje s obzirom na X Window System, a to je da formati prikaza između klijenta i poslužitelja moraju biti kompatibilni. Oba pristupa pokazuju da je propusnost mreže ograničavajući faktor.

Slično rješenje po imenu Chromium Renderserver [33] umjesto OpenGL-a koristi vlastitu biblioteku 100% kompatibilnu sa OpenGL-om koja omogućava paralelno iscrtavanje na više poslužitelja. Rezultirajuće slike mogu imati veću rezoluciju, npr. 4800x2400. Zaključak je da je ograničavajući faktor VNC posrednički (engl. *proxy*) poslužitelj, što je istovjetno rezultatima prethodno navedenog istraživanja.

Razvijeno je rješenje koje ovisno o traženoj kvaliteti, volumno iscrtavanje radi lokalno ili daljinski. U slučaju da je dovoljno korištenje odsječaka poravnatih s objektom (lošija kvaliteta), iscrtavanje se radi lokalno. Ako se želi koristiti odsječke poravnate s pogledom (viša kvaliteta), iscrtavanje se radi tako da se radiološki zapis prenese na udaljenu radnu stanicu, tamo iscrta i rezultirajuću sliku vrati natrag. Testiranja obavljena preko lokalne bežične mreže propusnosti

*GLX protokol je dodatak za X Window System koji se koristi na Unix i sličnim sustavima za prikaz grafičkog sučelja

34 Mb/s pokazuju da je moguće slike rezolucije 704x576 prenijeti 2-3 puta u sekundi [34]. U slučaju da se dimenzije slike smanje 4 puta, moguće je prenijeti do 15 slika u sekundi, ali dolazi do značajnog pada kvalitete. Slično istraživanje [35], usmjereno prvenstveno na kvalitetu prikaza, potvrđuje navedene zaključke.

2.9 Sustavi za iscrtavanje velikih količina podataka

Istraživanja iz područja radio astronomije generiraju velike količine podataka. Radi se o stotinama GB, pa do 1 TB. Takvu količinu podataka nije moguće obraditi na jednom računalu, već je potrebno korištenje grozda računala (engl. *cluster*). Razvijen je radni okvir [36] koji je testiran na 128 grafičkih procesora. Uz količinu podataka od 204 GB, postignut je prikaz od 30 slika u sekundi.

2.10 Povratne informacije

Provedeno je istraživanje o korištenju haptičkog uređaja s povratnom informacijom. Pretpostavka je bila da miš i tipkovnica nisu prikladno rješenje za potrebe planiranja operacija. Provedeno je testiranje na segmentaciji skena jetre i zaključeno je da dolazi do poboljšanja točnosti. Preporuka je da je potrebno pronaći ravnotežu između visoke cijene uređaja i naprednih mogućnosti koje takvi uređaju donose [37].

Istraženo je korištenje sučelja za virtualnu stvarnost za prikaz DICOM snimki u web pregledniku. Klijentsko računalo učitava web stranicu, spaja se na repozitorij DICOM datoteka, dohvaća DICOM datoteku te je prikazuje unutar web preglednika korištenjem X3DOM razvojnog okvira. Razvojni okvir X3DOM koristi WebGL i WebVR tehnologije unutar web preglednika za prikaz volumena. Ustanovljeno je da su postojeće tehnologije za virtualnu stvarnost orijentirane na to da korisniku daju osjećaj uronjenosti u 3D scenu. Prikazi volumena su plutali u prostoru pa ih se trebalo ograditi kockom. Osim toga bilo je naglih prelaza u sceni. Zaključeno je da ima mjesta za napredak, kako u web tehnologijama, tako i u uređajima [38].

2.11 Ograničenja mreže

Rješenje temeljeno na splet računarstvu (engl. *grid*), GridFTP protokolu i Globus Tookit biblioteci prikazano je na konferenciji iGrid2002. Ostvarena je veza između USA i Nizozemske koja je imala visoku propusnost i visoku latenciju. Korisnici su međusobno komunicirali preko videa i imali su mogućnost interaktivno manipulirati 3D objektima koji su se udaljeno iscrtavali. Pokazalo se da algoritam za kontrolu zagušenja TCP protokola ne radi dobro s vezama koje

imaju visoku propusnost i visoku latenciju. Preliminarni rezultati pokazali su da UDP protokol nema takav problem [39].

Računarstvo na rubu oblaka (engl. *edge computing*) obično se sastoji od tri razine: korisnika, kontrolera na rubu oblaka i oblaka (engl. *cloud*). Očekuje se da će računarstvo na rubu oblaka zajedno sa 5G tehnologijom podržavati puno bitnih rješenja kao što su upravljanje prometom, nadzor i praćenje zdravstvenog stanja. Isprobavanjem postojećih rješenja dolazi se do zaključka da postoje brojna ograničenja po pitanju cijene, potrošnje energije, hlađenja, kao i nepredvidivog ponašanja mreže. Također je uočen značajan pad performansi kod posluživanja velikog broja malih DICOM datoteka u odnosu na posluživanje malog broja velikih DICOM datoteka [40].

Poglavlje 3

Model učinkovite programske potpore i procedure za višekorisničku udaljenu vizualizaciju volumnih medicinskih podataka

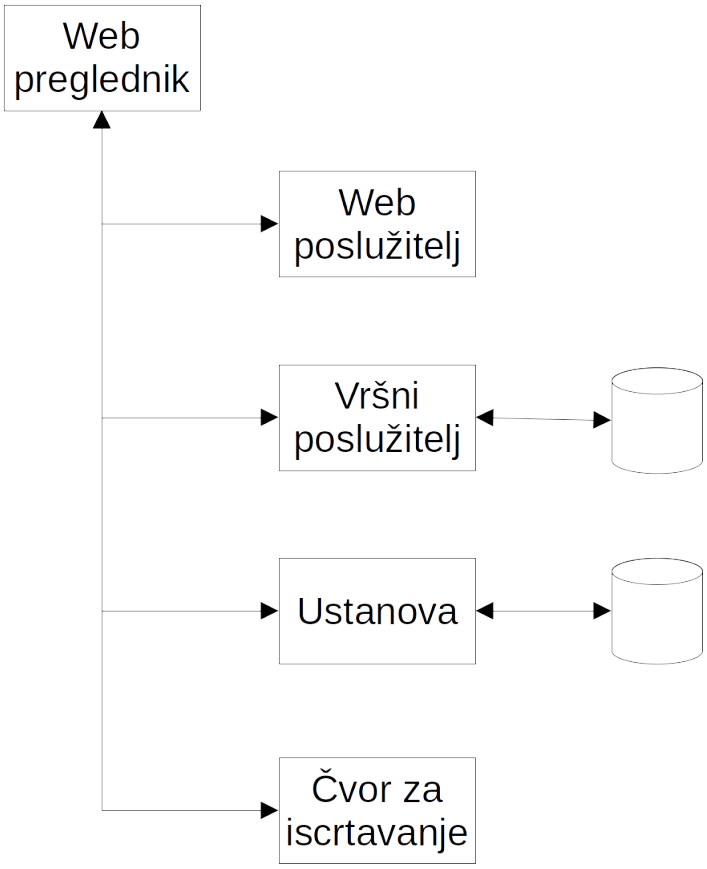
S motivacijom za rješavanje problema iz prakse (poglavlje 1) provedeno je istraživanje područja (poglavlje 2) te su definirani sljedeći zahtjevi za sustav:

- •Snimke se čuvaju u ustanovi gdje su nastale
- U slučaju da se radi o manjoj ustanovi ili o ustanovu u ruralnom području, rješenje treba biti takvo da se može pokrenuti na jednom računalu
- •Rješenje treba biti takvo da se može skalirati na podru čje cijele države
- •Snimke trebaju biti dostupne što prije nakon skeniranja
- •Prijenos podataka treba biti siguran
- Podržati novije verzije HTTP protokola jer prva verzija HTTP protokola ima nasljedna ograničenja zbog korištenja TCP-a (loša kontrola zagušenja)
- Korisnik sustavu pristupa preko web preglednika bez bilo kakve instalacije dodatnih programa
- Koristi se isključivo HTML5, CSS i JavaScript bez WebGL-a jer su korisnička računala pod-optimalna
- Podržana su stolna računala i tableti s velikim zaslonima jer analiza radioloških snimki na mobitelima nije realna
- •Nema potrebe za prijenosom ve ćeg broja slika u sekundi
- •O čuvanje kvalitete slike
- Minimizira se količina podataka koji se prenose zbog brzine i cijene usluge korištenja Interneta

- •Maksimalno se iskorištava postoje će sklopovlje
- •U slu čaju potrebe za investicijom u novu opremu, ona treba biti minimalna
- •Korisnik ne smije osjetiti latenciju sustava prilikom korištenja
- Koriste se postojeće tehnologije i norme, bez inovacije novih formata zapisa ili komunikacijskih protokola
- •Ne koriste se kola čići (engl. *cookies*) za pohranu podataka o sjednici

S ovako definiranim zahtjevima za sustav prvo je napravljen model sustava i definirana je komunikacija između dijelova sustava (poglavlje 3). Na osnovu dobivenog modela izgrađen je sustav na osnovu četveroslojne arhitekture (poglavlje 4). Prilikom izrade modela sustava potrebno je paziti na ponavljajući trend razvoja tehnologija i porasta veličine zapisa, tj. napraviti rješenje koje ne treba značajno prerađivati nakon određenog vremena.

3.1 Apstraktni model sustava



Slika 3.1 – Apstraktni model sustava

Sustav je zamišljen kao modularna četveroslojna struktura prikazana na slici 3.1. Sustav je izgrađen u cijelosti, ali je konstrukcija takva da se mogu izgraditi samo dijelovi sustava i integrirati u postojeći sustav koji se koristi u određenoj ustanovi. Komunikacija između dijelova

sustava detaljno je opisana pa se određeni dio sustava može napraviti kao priključak (engl. *plugin*) koji prevodi komunikaciju ovog sustava u komunikaciju nekog drugog sustava.

3.1.1 Web poslužitelj

Prva razina je web poslužitelj čija je jedina uloga poslužiti korisniku HTML, CSS i JavaScript datoteke. Interakcija korisnika sa sustavom počinje nakon učitavanja web stranice i prijave u sustav. Web stranica sastoji se od malog broja datoteka i odmah se učitava u cijelosti radi izbjegavanja brojnih zahtjeva i odgovora između web preglednika i poslužitelja. Kako se odmah učitava cijela stranica korisnik tijekom rada ne mora čekati na daljnja učitavanja. Na ovaj način se bilo kakva promjena na stranici događa unutar nekoliko milisekundi te korisnik ima osjećaj da je stranica vrlo responzivna.

Korisnik se u sustav prijavljuje korisničkim imenom i lozinkom. Lozinka se prije slanja na poslužitelj kriptira MD5 algoritmom. Lozinka je i u bazi podataka pohranjena kao MD5 kriptirani niz znakova radi zaštite korisnika. Korisnik možda istu lozinku koristi na više uređaja ili sustava koje koristi pa bi se pohrana lozinke u originalnom obliku mogla zloupotrijebiti.

Ovisno o ovlastima korisnika, dio izbornika na web stranici se skriva. Npr. ako korisnik nije administrator ne nude mu se izbornici za upravljanje korisničkim podacima.

3.1.2 Vršni poslužitelj

Vršni poslužitelj predviđen je za uporabu na razini države i čuva podatke koji moraju biti centralizirani. Uloge vršnog poslužitelja su:

- •Upravljanje podacima pacijenata
- •Upravljanje korisni čkim podacima (liječnicima i istraživačima iz područja medicine)
- •Upravljanje ustanovama
- •Prijavom korisnika u sustav
- •Administracijom vlastitih korisni čkih podataka
- •Upravljanje vrstama snimki koje sustav podržava
- •Obnova tokena za provjeru autenti čnosti

Svi podaci na ovoj razini pohranjuju se u bazu podataka. Svaka aktivnost na poslužitelju upravljana je sučeljem web stranice, osim obnove tokena. Token se koristi umjesto kolačića jer ne ostaje trajno pohranjen na računalu i vremenski istječe pa ga se ne može zloupotrijebiti, osim u kratkom periodu nakon odjave korisnika sa sustava što ne daje dovoljno vremena potencijalnom napadaču da napravi ozbiljnu štetu.

Uspješnom prijavom korisnika u sustav, unutar web preglednika pokreće se mjerač vremena koji otkucava cijelo vrijeme dok je korisnik prijavljen na sustav i periodički traži od poslužitelja da mu pošalje obnovljeni token za provjeru autentičnosti. Token za provjeru autentičnosti prenosi

se kod svakog zahtjeva od strane korisnika prema bilo kojoj razini poslužitelja, osim prilikom prijave u sustav jer se tada stvara. Sadržaj i konstrukcija tokena opisani su u odjeljku 3.4. Token se obnavlja svakih 5 minuta. U slučaju da token nije obnovljen, vremenski istječe te se odbijaju svi zahtjevi od strane korisnika prema bilo kojoj razini sustava. Prilikom odjave sa sustava isključuje se mjerač vremena te token istječe unutar 5 minuta.

3.1.3 Poslužitelj na razini ustanove

Ovaj poslužitelj predviđen je za uporabu na razini ustanove i ima sljedeće uloge:

- •Upravljanje čvorovima za iscrtavanje
- •Pohranu metapodataka o DICOM snimkama
- •Pohranu radioloških nalaza

Svi podaci na ovoj razini pohranjuju se u bazu podataka.

Kako se ovakvi poslužitelji mogu nalaziti u malim ustanovama koje nemaju osoblje za održavanje računala, radi provjere ispravnosti poslužitelja, preko web stranice može se raditi prozivanje (engl. *ping*).

3.1.4 Čvor za iscrtavanje

Ovaj poslužitelj ima samo dvije uloge: upravljanje DICOM snimkama i iscrtavanje slika.

Ako se pohranjuje veća količina DICOM snimki, snimke ne moraju biti pohranjene direktno na ovom poslužitelji već mogu biti pohranjene na SAN ili NAS uređaje.

DICOM snimka se na poslužitelj prenosi preko web preglednika korisnika unutar ustanove. Prilikom snimanja zapisa izbjegava se višestruko čitanje. Npr. prilikom izračuna MD5 sume datoteke istovremeno se dohvaćaju metapodaci (odjeljak 3.6). Po dovršetku izračuna MD5 sume, datoteka se preimenuje u dobiveni MD5 niz znakova te se MD5 suma i metapodaci o zapisu vraćaju korisniku (više u odjeljku 3.2).

Poslužitelji ove vrste trebaju raspolagati specijaliziranim grafičkim sklopovljem, bilo diskretnim grafičkim karticama ili integriranim u centralni procesor. Također trebaju raspolagati velikom količinom radne memorije jer je prilikom učitavanja DICOM snimke potrebno do deset puta više memorije od zauzeća prostora zapisa na tvrdom disku. Paralelno učitavanje više DICOM snimki može zauzimati nekoliko desetaka GB radne memorije pa je potrebno što bolje upravljati radnom memorijom. Odmah nakon što se DICOM zapis učita i prebaci u OpenGL teksturu, potrebno je osloboditi memoriju zauzetu prilikom učitavanja. Eksplicitno oslobađanje radne memorije traje određeno vrijeme, ali to je zanemarivo s obzirom koliko traje otvaranje slike.

Kao drugu mjeru potrebno je implementirati mjerač vremena koji oslobađa zauzetu memoriju. U slučaju da je korisnik zaboravio zatvoriti sliku unutar web preglednika ili je zatvorio web preglednik bez odjave sa sustava OpenGL tekstura i dalje zauzima memoriju. Mjerač vremena

će nakon 5 minuta od zadnje interakcije korisnika sa slikom osloboditi memoriju tako da obriše OpenGL teksturu.

Kako ovakvih poslužitelja može biti više unutar ustanove ili se mogu nalaziti u malim ustanovama koje nemaju osoblje za održavanje računala, radi provjere ispravnosti poslužitelja preko web stranice se može raditi prozivanje.

3.2 Komunikacija između elemenata sustava

Komunikacija između web stranice i poslužitelja opisana je po OpenAPI specifikaciji [41]. Specifikacija komunikacijskog sučelja može se zapisati u JSON ili YAML obliku zapisa. U izgradnji modela sustava korišten je YAML oblik zapisa [42] (vidi primjer u dodatku A).

Prilikom izgradnje modela, prvo se definiralo komunikacijsko sučelje, a zatim komponente sustava. Komunikacija između elemenata sustava zamišljena je tako da se početak uvijek inicira od strane korisnika preko web sučelja. Komunikacija ide uvijek direktno između web stranice i određenog poslužitelja. Komunikacije između poslužitelja nema niti na istoj hijerarhijskoj razini ni između hijerarhijskih razina. Komunikacija između poslužitelja nije potrebna jer se sve potrebne informacije za sve zahtjeve od strane korisnika prema bilo kojem poslužitelju nalaze u tokenu za provjeru autentičnosti. Na ovaj način se teret velikog broja upita/odgovora prenosi s poslužitelja viših hijerarhijskih razina (pogotovo vršnog poslužitelja) na računalo krajnjeg korisnika. Iako su ta računala pod-optimalna za iscrtavanje grafike, nisu za komunikaciju. Komunikacija se u najgorem slučaju pretraživanja (pretraživanje po svim ustanova) svodi na N zahtjeva i N odgovora. Svaki zahtjev i odgovor poruke su male veličine koje ne predstavljaju veliku količinu podataka na vezi čije se korištenje naplaćuje, niti značajno opterećuje resurse računala koje inicira ovakav oblik komunikacije. Međutim, značajno se rasterećuje vršni poslužitelj koji bi inače morao obraditi 2xNxM (M je broj korisnika sustava u određenom trenutku) poruka.

Korišteni oblici komunikacije mogu se podijeliti u sljedeće grupe:

1.Trivijalni - prijava u sustav, odjava, promjena lozinke, obnova tokena

2. Uobičajeni zahtjev na osnovu kojega poslužitelj obavi neku radnju i vrati odgovor - upravljanje korisnicima, pacijentima, ustanovama ...
3. Višerazinski - učitavanjem DICOM zapisa na čvor za iscrtavanje kao odgovor dobiju se metapodaci o zapisu. Ti metapodaci šalju se poslužitelju na razini institucije koji ih pohranjuje.
4. Pretraživanje zapisa o očitanjima DICOM zapisa - šalju se zahtjevi na sve institucije po određenom kriteriju pretraživanja (pacijent, vremenski raspon ...)

5.U čitavanje DICOM zapisa i iscrtavanje slike - modificirana izvedba oblika 2

U četvrtom slučaju rasterećuje se vršni poslužitelj koji bi inače po zahtjevu korisnika poslao upit na sve poslužitelje na razini institucije, prikupio odgovore, objedinio ih i poslao korisniku. Ovisno

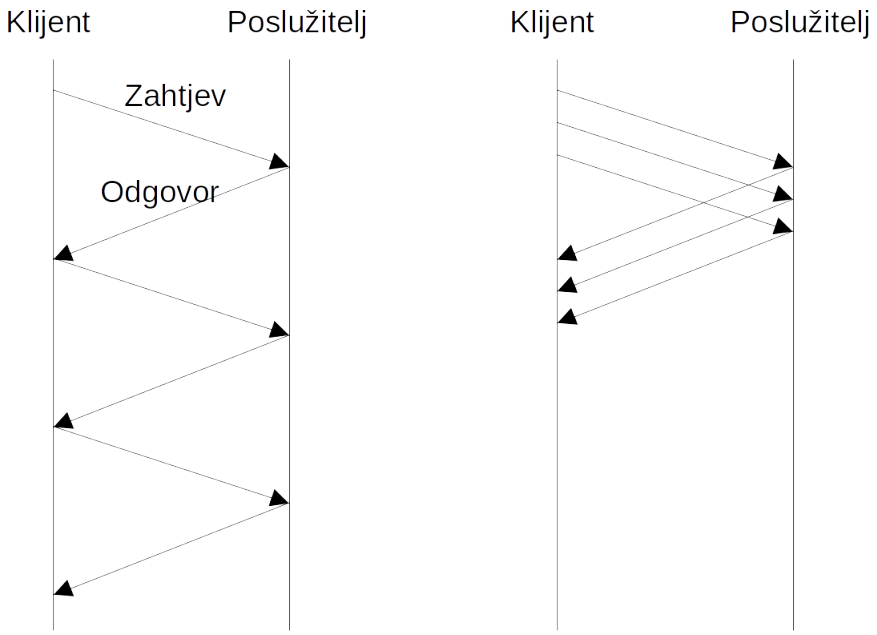
o propusnosti mreže i opterećenju pojedinog poslužitelja, odgovor može potrajati nekoliko sekundi. To znači da nakon klika na gumb za pretragu korisnik čeka određeno vrijeme dok ne dobije cjeloviti odgovor. Ako se ova komunikacija prebaci na računalo korisnika, korisnik ima privid da je sustav responzivniji jer se dio odgovora počinje pojavljivati vrlo brzo, a kasniji odgovori dinamički dopunjuju listu rezultata pretraživanja. Ukupna duljina čekanja je nešto kraća jer se izbjegla komunikacija s vršnim poslužiteljem, a subjektivni dojam korištenja sustava je pozitivniji.

Kod prva četiri oblika komunikacije pošalje se zahtjev, pričeka na obradu te se prima odgovor. Kod petog oblika pristup je drugačiji. Učitavanje DICOM snimke, u slučaju zapisa veličine nekoliko GB, može trajati nekoliko desetaka sekundi. Korisnik već nakon 15 sekundi gubi koncentraciju [5] što utječe na kvalitetu njegova rada. Zbog toga se u ovom slučaju ne čeka na dovršetak učitavanja cijele snimke nego se korisniku vraća odgovor odmah nakon što je datoteka uspješno otvorena. Korisnik može odmah početi pregledavati snimku. Tijekom učitavanja DICOM snimke za svaki sloj bilježi se da li je učitan ili ne te se po dovršetku učitavanja cijela snimka označava kao učitana. Korisnik klizačem na web sučelju odabire sloj snimke koji želi pogledati. Ako odluči pogledati prvi sloj, može ga odmah pogledati jer je prvi (i još nekoliko sljedećih slojeva) učitan do trenutka kada je do korisnika došao odgovor o učitavanju snimke. U slučaju da je korisnik odabrao neki od slojeva koji još nije učitan, dobiva poruku da je učitavanje još u tijeku. Akcija korisnika pri kojoj odabire položaj klizača i klika na gumb za dohvat slike traje nekoliko sekundi, a do tada je DICOM zapis veličine do nekoliko stotina MB već učitan pa je velika vjerojatnost da korisnik uopće neće dobiti poruku da učitavanje još traje. Na ovakav način skoro je u potpunosti prikrivena latencija sustava prilikom pregledavanja DICOM snimki i korisnik ima osjećaj trenutnog odziva sustava. Kod volumnog iscrtavanja latencija ne može se u potpunosti prikriti jer iscrtavanje volumena ne može početi dok cijela snimka nije učitana (inače dolazi do djelomičnog iscrtavanja volumena i daje se kriva informacija korisniku što se ne smije dogoditi u slučaju medicinske dijagnostike). U ovom slučaju latencija je prikrivena u većoj mjeri i korisnik možda mora čekati nekoliko sekundi na pojavu rezultata iscrtavanja volumena.

3.3 Nove verzije HTTP protokola

Zbog ranije spomenutih nedostataka HTTP i HTTPS protokola, podržano je korištenje samo HTTPS protokola s najnovijim verzijama TLS protokola. Osim toga podržan je i HTTP/2 protokol zbog svojih naprednih mogućnosti [43]:

- Komprimiranje HTTP zaglavlja
- Slanje zahtjeva bez čekanja na prethodni odgovor
- Multipleksiranje više zahtjeva preko jedne TCP veze
- Kriptiranje



Slika 3.2 – Slanje zahtjeva sa i bez čekanja na prethodni odgovor

Također je podržan i HTTP/3 protokol [44]. HTTP/3 umjesto TCP protokola koristi QUIC. QUIC je protokol povrh UDP protokola koji rješava problem kontrole zagušenja [45]. Latencija protokola je manja jer u slučaju gubitka paketa, za razliku od TCP-a koji ponovno traži paket i čeka dok se paket ne isporuči, QUIC traži paket ali ne čeka nego nastavlja prijenos dok novi ispravni paket ne stigne. HTTP/3 protokol skraćuje trajanje rukovanja (engl. *handshaking*) prilikom uspostave veze i dodatno povećava sigurnost.

3.4 Token za provjeru autentičnosti

Umjesto kolačića koriste se JWT tokeni [46]. Token se generira prilikom prijave korisnika u sustav i periodički obnavlja mjeračem vremena koji je pokrenut unutar web preglednika. Svaki zahtjev koji ide korisnika prema bilo kojoj razini poslužitelja (osim web poslužitelja koji se više ne koristi nakon učitavanja web stranice) nosi JWT token u zaglavlju HTTP poruke.

Poslužitelj nakon primitka HTTP poruke prvo provjerava token. Provjerava se da li je token istekao, da li je korisnik ovlašten za pristup REST ruti kojoj pokušava pristupiti te zaštitna suma JWT tokena.

Token prenosi sljedeće podatke:

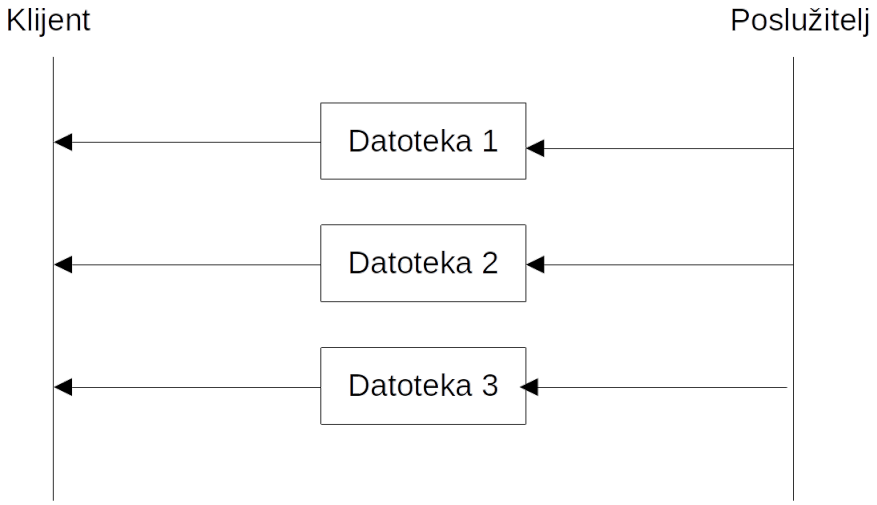
•Da li je korisnik administrator

•Da li je korisnik super-korisnik

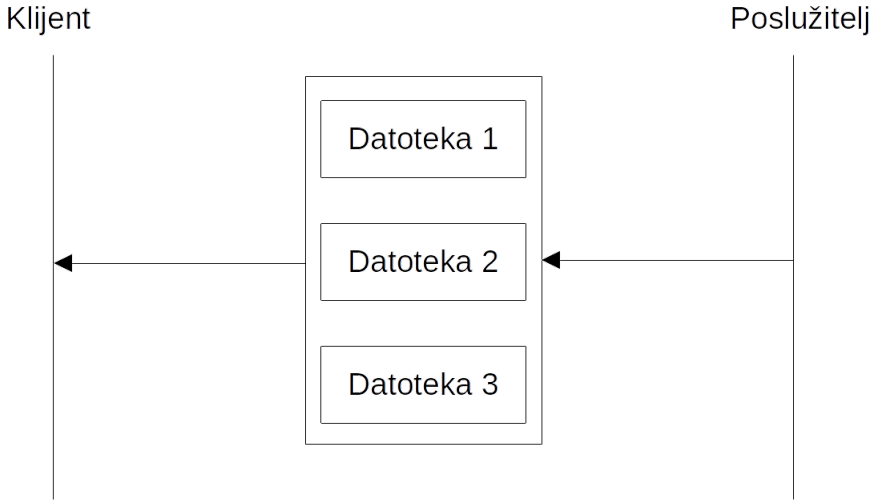
•Da li je korisnik lije čnik

•Jedinstveni identifikator korisnika

•Jedinstveni identifikator ustanove u kojoj korisnik radi



Slika 3.3 – Prijenos bez multipleksiranja veze



Slika 3.4 – Prijenos sa multipleksiranjem veze

- •IP adresa poslužitelja na razini ustanove
- •Port poslužitelja na razini ustanove
- •Datum i vrijeme stvaranja tokena
- •Vrijeme trajanja tokena

Token se sastoji od tri dijela:

- •Zaglavlja koje definira algoritam i vrstu tokena
- •Korisnih podataka (gornja lista)
- •Zaštitne sume

Zaštitna suma generira se na osnovu zaglavlja, korisnih podataka i lozinke. Nakon toga se zaglavlje i korisni podaci kodiraju base64UrlEncode algoritmom, poredaju u niz odvojen točkom i na kraju doda zaštitna suma odvojena točkom. Lozinka se nalazi na poslužitelju i nikada se ne prenosi preko mreže. Praćenjem prometa između web preglednika i bilo kojeg poslužitelja

```
Zaglavlje:
{
        "alg": "HS256",
        "typ": "JWT"
}

Korisni podaci:
{
        "first_name": "Krešimir",
        "last_name": "Jozić",
        "iat": 1516239022
}

Zaštitna suma:
HMACSHA256(
        base64UrlEncode(header) + "." +
        base64UrlEncode(payload),
        lozinka
)
```

Slika 3.5 – Primjer JWT tokena prije kodiranja

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmaXJzdF9uYW1lIjoiS3JlxaFpbWlyIiwibGFzdF9u
YW1lIjoiSm96acSHIiwiaWF0IjoxNTE2MjM5MDIyfQ.qDQsUgxzkNs94LMn21qURd6xUbH9_4
AIsrQtXGsWYcQ
```

Slika 3.6 – Primjer JWT tokena kodiranog lozinkom „lozinka”

nije moguće saznati lozinku, promijeniti token i generirati novu zaštitnu sumu. Drugim riječima, ako se netko uspije domoći tokena (što je iznimno teško, ali ipak moguće ako se koristi SSL ili starije verzije TLS protokola [47]) i izmjeni korisne podatke, neće doći do povećanja razine ovlasti korisnika jer zaštitna suma neće odgovarati i poslužitelj će odbaciti takav token.

U slučaju da lozinka ipak bude kompromitirana, implementiran je mehanizam za brzu promjenu lozinke bez zaustavljanja poslužitelja. Lozinku se zapisuje u datoteku s postavkama poslužitelja. U slučaju potrebe promjene lozinke, u datoteku s postavkama zapiše se nova lozinka i poslužitelj se slanjem signala SIGUSR1 obavijesti da treba učitati nove postavke.

3.5 Ograničenje pristupa

Svaka usluga koju bilo koji poslužitelj pruža korisniku preko web stranice definirana je određenom putanjom. Svaka putanja štiti se na dva načina. Prvi način je skrivanje izbornika ili gumba koji bi doveo do aktiviranja putanje, ovisno o ovlastima korisnika. U slučaju da korisnik poznaje

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmaXJzdF9uYW1lIjoiS3JlxaFpbWlyIiwibGFzdF9u
YW1lIjoiSm96acSHIiwiaWF0IjoxNTE2MjM5MDIyfQ.PEOnyN0BFSL3q8OGrqntnMEMLVkv41
vnztC-Hqm8244
```

Slika 3.7 – Primjer JWT tokena kodiranog lozinkom „secret”

HTML i JavaScript, mogao bi aktivirati putanju mimo web sučelja kroz razvojne alate web preglednika. Radi toga se putanja štiti JWT tokenom koji u sebi nosi sve potrebne podatke o korisniku i njegovim pravima. Primitkom HTTP poruke, na početku svake aktivnosti na poslužitelju provjerava se zaštitna suma tokena, da li je istekao te da li korisnik ima pravo na uslugu koju traži od poslužitelja. Neke putanje se ne štite, npr. prijava korisnika u sustav. Postoje i putanje koje za aktivnost koju obavljaju ne trebaju sadržaj HTTP poruke nego samo token, npr. obnova tokena.

3.6 DICOM snimke

Struktura zapisa DICOM snimki sastoji se od nekoliko dijelova:

- •Zaglavlje
  - – Preambula
  - – Prefiks
- •Skup podatkovnih elemenata
  - – Podatkovni elementi
- •Slojevi snimke

Preambula je veličine 128 bajta nakon čega slijedi prefiks „DICM”. Skup podatkovnih elemenata sastoji se od niza elemenata koji sadrže metapodatke o snimci, podatke o pacijentu, uređaju korištenom za skeniranje i još brojne druge podatke. Mogućnost definiranja ovolike količine podataka glavna je mana DICOM snimki jer se sadržaj velikog broja podatkovnih elemenata može proizvoljno tumačiti što dovodi do dvosmislenosti i zabuna. Podatkovni elementi su oblika: (XXXX, YYYY) neki_podatak. Primjer metapodataka jednostavnog DICOM zapisa može se pronaći u dodatku B.

Model ne prihvaća bilo koju DICOM snimku već je ograničen po parametrima danim u tablici 3.1.

Podatkovni element	Opis elementa	Značenje elementa	Ograničenja koja propisuje model
(0028,0002)	SamplesPerPixel	Broj kanala boje	1
(0028,0004)	PhotometricInterpretation	Vrsta slike	MONOCHROME1, MONOCHROME2
(0028,0010)	Rows	Broj redaka	>= 1
(0028,0011)	Cols	Broj stupaca	>= 1
(0028,0008)	NumberOfFrames	Broj slojeva slike	>= 1
(0028,0100)	BitsAllocated	Broj binarnih znamenki za pohranu elementa slike	16
(0028,1052)	RescaleIntercept	Presijecanje	nema
(0028,1053)	RescaleSlope	Nagib	nema
(0028,1050)	WindowCenter	Centar prozora	nema
(0028,1051)	WindowWidth	Širina prozora	nema
(0028,0103)	PixelRepresentation	Način zapisa cijelog broja	0, 1

Tablica 3.1 – Nužni metapodaci o DICOM snimci

Većinu parametara iz tablice nije potrebno objašnjavati osim RescaleSlope, RescaleIntercept, WindowCenter i WindowWidth.

Elementi CT snimki obično zauzimaju 12 bita, a čuvaju se u 16 bita od kojih je 4 neiskorišteno. Za prikaz CT snimki koristi se Hounsfield skala. To je kvantitativna je ljestvica za opis radiodenziteta. Pretvorba vrijednosti elemenata CT snimke u Hounsfield skalu radi se po formuli $HU = CTvrijednost * RescaleSlope + RescaleIntercept$. Uobičajene vrijednosti su $RescaleSlope = 1$ i $RecaleIntercept = -1000$ koje preslikavaju interval CT vrijednosti [0, 4095] u interval [-1000, 3095].

Preostala dva parametra ograničavaju vrijednost svjetline. DICOM snimke gledane bez obrade ispadaju pretamne ili presvijetle. Radi toga se ograničava raspon svjetlina po donjim formulama.

$$ako\ x <= c - \frac{1}{2} - \frac{w-1}{2},\ onda\ y = y_{min}$$

$$inače\ ako\ x > c - \frac{1}{2} + \frac{w-1}{2},\ onda\ y = y_{max}$$

$$inače\ y = \left(\frac{x - \left(c - \frac{1}{2}\right)}{w-1} + \frac{1}{2} \right) * (y_{max} - y_{min}) + y_{min}$$

gdje su: x = ulazna vrijednost, y = izlazna vrijednost, c = centar prozora (WindowCenter), w = širina prozora (WindowWidth)

3.7 Pohrana podataka

Za pohranu DICOM zapisa potrebno je koristiti najnovije datotečne sustave, točnije ZFS [48]. Datotečni sustav ZFS nastao je 2001 g. kao dio operacijskog sustava Solaris tvrtke Sun Microsystems. Programski kod datotečnog sustava bio je otvoren kao dio operacijskog sustava OpenSolaris od 2005. do 2010. g., kada je ponovno zatvoren gašenjem OpenSolaris-a kupnjom tvrtke Sun Microsystems od strane tvrtke Oracle. Tijekom tih 5 godina, datotečni sustav je prenesen na operacijske sustava Linux, Mac OS X i FreeBSD. 2013. g. osnovana je zaklada OpenZFS, s ciljem da nastavi njegov razvoj kao projekt otvorenog koda. 2020. g. OpenZFS obuhvatio je programski kod operacijskih sustava Linux i FreeBSD te ujedinio napore oko daljnjeg razvoja.

ZFS je odabran zbog brojnih naprednih mogućnosti koje drugi datotečni sustavi nemaju ili ih imaju u ograničenom obliku. Neke od tih mogućnosti su:

- Programska RAID polja - otporna na skrivena oštećenja podataka koja su karakteristična za sklopovske RAID kontrolere

- •Sažimanje
- •Kriptiranje
- •Lagana provjera integriteta podataka i popravak ošte ćenih podataka
- Pohranu zapisa u dnevniku prije potpunog zapisivanja za slučaj naglog nestanka električne energije (slično kao kod baza podataka)
- •Deduplikacija podataka
- •Razli čite strategije čuvanja priručnih podataka u cilju poboljšanja performansi
- Brojne mogućnosti podešavanja omogućavaju optimalno korištenje na diskovima različitih karakteristika
- Mogućnost izrade sigurnosne kopije podataka po principu kopiraj-prilikom-zapisivanja (engl. *copy-on-write*)

Posebno je zanimljiva mogućnost kriptiranja podataka jer sustav dobiva mogućnost zaštite podataka prilikom pohrane, a ne samo prilikom prijenosa.

3.8 OpenGL

OpenGL je aplikacijsko programsko sučelje za računalnu grafiku koje je napravila tvrtka Silicon Graphics 1992. g. Od 2006. g. održava ga i unaprjeđuje Khronos grupa [49]. OpenGL je trenutno najraširenije sučelje za računalnu grafiku jer je podržano na većini operacijskih sustava i programskih jezika. OpenGL se temelji na složenom automatu stanja (nekoliko stotina stanja) i implementira se unutar upravljačkog programa grafičkog sklopovlja. Nedostatak mu je to što se izvodi samo na jednoj dretvi te radi toga ne iskorištava prednosti paralelnosti izvođenja na višejezgrenim procesorima.

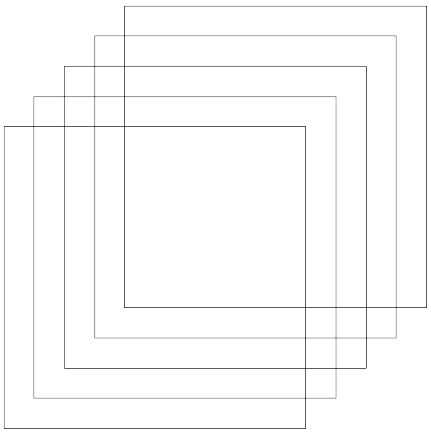
OpenGL kontekst povezuje automat stanja i prozor u kojemu se iscrtava grafika. Taj prozor može biti ekran (ili dio ekrana) na zaslonu korisnika ili spremnik u memoriji grafičkog sklopovlja. OpenGL se koristi tako da se aktivira kontekst, uz njega veže neki objekt (npr. tekstura, program za sjenčanje) te pošalju naredbe za iscrtavanje. Konteksta može biti više istovremeno, ali samo jedan može biti aktivan u određenom trenutku. Kako se OpenGL koristi u čvoru za iscrtavanje kojemu pristupa više korisnika istovremeno, stvara se jedan kontekst po korisniku. Pristup kontekstima štiti se mutex-om. Kako unutar istog konteksta može biti više istovremenih zahtjeva za određenom operacijom, npr. prilikom učitavanja DICOM zapisa u teksturu, operacije unutar istog konteksta također se štite mutex-om. Zaštita konteksta i operacija unutar konteksta štite se istim mutex-om jer samo jedan kontekst može biti aktivan u određenom trenutku, kao i pristup istoj teksturi unutar jednog konteksta. Ovakva zaštita dovodi do čekanja dretvi koje su spremne, npr. sljedeći sloj DICOM zapisa je učitan, ali prethodni još nije spremljen u teksturu. S obzirom na to da se OpenGL izvodi na jednoj dretvi drugačija izvedba nije moguća.

Detaljan opis rada OpenGL-a premašuje okvire ovoga rada te će biti spomenuti samo dijelovi

bitni za ovaj rad.

Put transformacija podataka od ulaznih koordinata točaka do izlazne slike definiran je OpenGL cjevovodom. Ulaz u cjevovod su koordinate vrhova geometrijskih tijela (trokuta), nakon čega slijedi program za sjenčanje vrhova (engl. *vertex shader*), razbijanje ili grupiranje geometrijskih tijela, program za sjenčanje geometrije, skupljanje geometrijskih tijela, rasterizacija, program za sjenčanje elemenata (engl. *fragment shader*) slike te još nekoliko faza testiranja i transformacija elemenata slike. Za ovaj rad najbitniji su programi za sjenčanje vrhova i programi za sjenčanje elemenata slike. Programi za sjenčanje pišu se u programskoj jeziku GLSL koji je temeljen na programskom jeziku C. Programi se prevode u strojni oblik i izvode na mnogobrojnim jezgrama (stotine ili tisuće) grafičkog sklopovlja.

Drugi bitan dio OpenGL-a su teksture. Teksture su objekti koji se sastoje od jedne ili više slika istog formata. Osim slika sadrže opise načina uzorkovanja, filtriranja kod povećanja ili smanjenja te načina prelaska preko rubova slika. Dimenzionalnost može biti 1, 2 ili 3. Osim jednostavnih oblika postoje još i polje 1D tekstura, polje 2D tekstura i stranice kocke.

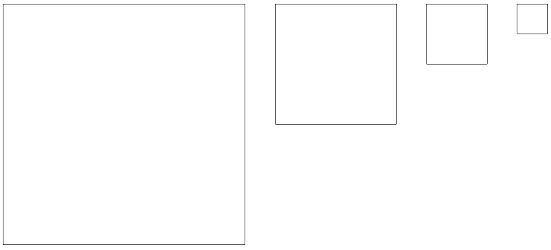


Slika 3.8 – Polje 2D tekstura

Dimenzije tekstura u ranijim verzijama OpenGL-a bile su ograničene na potencije broja 2 dok kod novijih verzija tog ograničenja nema. Maksimalne dimenzije značajno variraju ovisno o dimenzionalnosti teksture. Npr. 2D teksture mogu imati maksimalne dimenzije 16384x16384 dok 3D teksture mogu imati dimenzije 2048x2048x2048, ovisno o implementaciji.

Svaka tekstura, odnosno svaki sloj teksture može imati MIP (*lat. Multum In Parvo*) mape. MIP mape su nizovi umanjenih slika osnove teksture gdje svaki sljedeći element niza ima dimenzije 50% prethodnog. Niz se nastavlja do konačnog elementa koji ima dimenzije 1x1. Svi do sada nabrojani oblici tekstura podržavanju MIP mape. Ukupno zauzeće memorije teksture koja koristi MIP mape je 30% veće.

Model podržava maksimalno 4 razine MIP mapa. Ako su početne dimenzije 2048x2048 krajnja je 256x256, a tu je već toliko velik gubitak kvalitete da se ionako ne može koristiti za dijagnostiku. MIP mape u modelu se prvenstveno koriste za povećanje kvalitete slike



Slika 3.9 – MIP mapa

prilikom filtriranja. Filtriranje kod smanjenja ili povećanja može biti odabirom točke najbliže mjestu uzorkovanja ili linearnom interpolacijom najbližih susjednih točaka. Kod smanjenja slike događaju se artefakti koji dovode do gubitka kvalitete slike pa se može koristiti i tzv. trilinearno filtriranje koje koristi linearno filtriranje na sloju teksture i linearnu interpolaciju između susjednog elementa MIP mape.

U zadnjih nekoliko verzija OpenGL-a primjenjuje se AZDO (engl. *Approaching Zero Driver Overhead*) pristup. Kako je OpenGL temeljen na složenom automatu stanja, prijelazi između stanja na razini upravljačkog programa troše vrijeme na održavanje automata stanja umjesto na iscrtavanje. Čest način upravljanja objektima u OpenGL-u je takav da se prvo kreira objekt, npr. tekstura, veže uz OpenGL kontekst i onda koristi za neku operaciju. Od verzije 4.5 OpenGL moguć je direktan pristup objektima preko imena, a ne preko vezanja uz kontekst čime se skraćuje vrijeme vezano uz održavanje automata stanja i ostaje više vremena za korisne operacije. Drugi pristup koji se koristi u ovom radu je rezervacija memorije za teksture. Kod starijih verzija tekstura, prvo se definirao format pa su se učitavali slojevi teksture - odmah svi ili jedan po jedan. Moglo se dogoditi da dimenzije nisu iste, iako trebaju biti, a to se otkrilo tek prilikom iscrtavanja. Kod novijih verzija OpenGL-a odmah se zauzima prostor za sve slojeve teksture i prilikom učitavanja se provjerava da li odgovaraju zadanom formatu. Na ovaj način kasnija provjera prilikom iscrtavanja nije potrebna čime se poboljšavaju performanse. Osim navedenih pristupa još nekoliko pristupa sačinjavaju AZDO, ali nisu bitni za ovaj rad pa neće biti opisani.

3.9 Iscrtavanje slika

Model podržava prikaz pojedinih slojeva zapisa i iscrtavanje volumena na osnovu korištenja 3D tekstura. Zbog toga se ograničavaju dimenzije ulaznih DICOM zapisa koji se pohranjuju u teksture dimenzija koje su maksimalno 2048x2048x2048 (maksimalne dimenzije ograničene su korištenim grafičkim sklopovljem).

Ako ulazni zapis ima više od 2048 slojeva on se odbacuje jer se ne može pohraniti u teksturu. Ako se učita nekompletan zapis, može doći do gubitka informacija ključnih za dijagnozu. Ako su dimenzije pojedinog sloja veće od 2048x2048 one se smanjuju na dimenzije 2048x2048 uz zadržavanje omjera. Ako npr. ulazni zapis ima dimenzije 1600x2500, smanjuje se na dimenzije

```
#version 460

layout(location = 0) in vec3 vertCoord;
layout(location = 1) in vec2 vertTexCoord;
out vec2 fragTexCoord;

void main()
{
    gl_Position = vec4(vertCoord.x, vertCoord.y, 0.0, 1.0);
    fragTexCoord = vertTexCoord;
}
```

Slika 3.10 – Program za sjenčanje vrhova ulazne teksture

1311x2048.

Proces učitavanja snimke izvodi se tako da se učita jedan sloj DICOM zapisa u ulaznu teksturu. Dimenzije teksture odgovaraju dimenzijama ulaznog zapisa, kao i broj bitova (mora biti 16) te nemaju informacije o boji nego samo o svjetlini. Ulazna tekstura se programima za sjenčanje vrhova (slika 3.10) i elemenata slike (slika 3.11) preslikava u teksturu za pohranu cijelog zapisa. Kada korisnik zatraži određeni sloj slike ili cijeli volumen, tekstura za pohranu se programima za sjenčanje vrhova (isti kao za ulaznu teksturu) i elemenata slike (slika 3.12) preslikava u izlaznu teksturu koja je dimenzija istih kao i dimenzije okvira unutar web preglednika, ali s očuvanim omjerom stranica originalne teksture. Kako se OpenGL kontekst štiti mutexima, proces iscrtavanja izlazne slike može se odvijati dok proces učitavanja još traje, osim u slučaju iscrtavanja volumena kada su potrebni svi slojevi teksture. Dok traje učitavanje DICOM zapisa, filtriranje teksture za pohranu podešeno je na linearno. Po dovršetku učitavanja generiraju se MIP mape i filtriranje postavlja na trilinearno.

U slučaju pregledavanja snimki, procedura učitavanja i prikaza slika svodi se na prilagodbu dimenzija i formata zapisa iz ulazne teksture u teksturu za pohranu pa potom u izlaznu teksturu. Ulazna tekstura ima elemente zapisane kao vrijednosti intenziteta u obliku 16 bitnih cijelih brojeva s predznakom. Preslikavanje se radi tako da se iscrtaju dva trokuta koja prekrivaju cijeli ekran (koji se ne prikazuju na ekranu nego se spremaju u memoriju), na njih se nalijepi tekstura i nakon rasterizacije nad svakim elementom slike obave transformacije programa za sjenčanje. Program za sjenčanje prilikom uzorkovanja teksture pretvara zapis u cijeli broj i normira ga u domenu [-1, 1]. Taj broj se pomnoži s 32768 da se dobije cijeli broj nad kojim se radi pretvorba u HU ljestvicu pa potom radi prilagodba svjetline centrom i širinom prozora. Rezultat je realni broj iz domene [0, 1] koji se sprema u teksturu za pohranu kao 8 bitni cijeli broj bez predznaka (domena je [0, 255]) [50]. Ove pretvorbe i normiranja su potrebne radi linearnog filtriranja. Slike se mogu uzorkovati i kao cjelobrojne ali u tom slučaju je dostupno samo uzorkovanje najbližeg susjeda bez linearne interpolacije. Filtriranje teksture za pohranu i preslikavanje u

```
#version 460

in vec2 fragTexCoord;
layout(location = 0) out float outputColor;
layout(binding = 0) uniform sampler2D texSampler;
layout(location = 0) uniform float rescaleSlope;
layout(location = 1) uniform float rescaleIntercept;
layout(location = 2) uniform float windowCenter;
layout(location = 3) uniform float windowWidth;

const uint out_min = 0;
const uint out_max = 255;

void main() {
    float sampled = texture(texSampler, fragTexCoord).r * 32768.0f;
    float ww = windowWidth - 1.0f;
    float wc = windowCenter - 0.5f;
    float hw = ww / 2.0f;
    float val = sampled * rescaleSlope + rescaleIntercept;

    if (val <= wc - hw) {
        outputColor = 0.0f;
    } else if (val > wc + hw) {
        outputColor = 1.0f;
    } else {
        outputColor = ((val - wc) / ww + 0.5f);
    }
}
```

Slika 3.11 – Program za sjenčanje elemenata slike ulazne teksture

```
#version 460

in vec2 fragTexCoord;
layout(location = 0) out float outputColor;
layout(binding = 1) uniform sampler3D texSampler;
layout(location = 0) uniform float depth;


void main() {
    outputColor = texture(texSampler, vec3(fragTexCoord, depth)).r;
}
```

Slika 3.12 – Program za sjenčanje elemenata slike izlazne teksture

izlaznu teksturu jednostavnije je jer su formati zapisa isti, samo što se u ovom slučaju uzorkuje 3D tekstura i radi trilinearna interpolacija pa je ova faza računski zahtjevnija.

Iscrtavanje volumena računski je puno zahtjevnije jer zahtijeva N uzorkovanja tekstura i akumulaciju uzorkovanih vrijednosti (formula 3.1).

$$y_N = y_{N-1} + (1 - y_{N-1}) * x \tag{3.1}$$

gdje su: y_N= trenutna akumulirana vrijednost, y_{N-1}= akumulirana vrijednost u prethodnoj iteraciji i x= uzorkovana vrijednost.

Na modernom grafičkom sklopovlju koriste se jednoprolazni i dvoprolazni algoritmi. Dvoprolazni algoritam prvo iscrtava kocku čije su stranice obojene crvenom, zelenom i plavom bojom tako da RBG komponente boje odgovaraju 3D koordinatama. Iscrtavaju se prednja i stražnja strana kocke iz perspektive korisnika i spremaju u pomoćne teksture. U drugom prolazu uzorkuju se pomoćne teksture i definira vektor koji predstavlja zraku emitiranu iz točke iz koje korisnik gleda u dio ekrana na kojemu se prikazuje slika. Na osnovu dobivenog vektora obavlja se uzorkovanje 3D teksture duž tog vektora. Uzorkovanje se obavlja u fiksnim intervalima sve dok zraka ne izađe izvan teksture ili dok vrijednost akumulirane svjetline ne dostigne unaprijed zadanu maksimalnu vrijednost [51], [52]. Jednoprolazni algoritam ne koriste pomoćne teksture za izračun vektora već to radi transformacijskim matricama i koordinatama pogleda korisnika [53], [54]. Način uzorkovanja 3D teksture isti je kao u dvoprolaznom algoritmu.

Poglavlje 4

Platformski neovisna arhitektura programske potpore zasnovana na vizualizacijsko-komunikacijskom radnom toku

Na osnovu modela i komunikacijskih procedura opisanih u prethodnom poglavlju konstruirana je arhitektura sustava (slika 4.1) te je implementirano programsko rješenje. Arhitektura sustava napravljena je tako da bude platformski neovisna. To znači da se treba moći pokrenuti na bio kojoj modernoj arhitekturi procesora te pod bilo kojim operacijskim sustavom koji ima podršku za OpenGL 4.5 ili 4.6. OpenGL 4.5 norma objavljena je 2014. g. što znači da sklopovlje koje se koristi ne smije biti starije od osam godina. S obzirom na to da je nepisano pravilo da poslužitelje treba mijenjati svakih pet godina, zahtjev od osam godina omogućava produljenje životnog vijeka sklopovlje te uštede korisnicima. Najvrjedniji dio sustava su podaci zapisani u baze podataka i DICOM zapisi. Način zaštite podataka u bazama opisan je dalje u ovom poglavlju, a za DICOM zapise preporučuje se pohrana na RAID poljima ili SAN/NAS uređajima.

Slika 4.1 – Hijerarhijski prikazana arhitektura sustava

4.1 Pokretanje sustava

Sustav je izveden tako da se u slučaju kvara poslužitelja ili instalacije novog poslužitelja lako preseli i pokrene. Instalacija nije potrebna, nego je dovoljno prekopirati potrebne datoteke i pokrenuti sustav. Cjelovito programsko rješenje sadržano je u jednoj izvršnoj datoteci. Osim nje, potrebno je kopirati pogon baze podataka (također jedna izvršna datoteka) te samu bazu podataka koja se sastoji od više datoteka u jednom direktoriju. Nakon kopiranja dvije datoteke i jednog direktorija potrebno je napisati (ili preraditi) postojeću konfiguracijsku datoteku zapisanu u YAML obliku. Preostaje samo pokrenuti bazu podataka te nakon nje izvršnu datoteku sustava.

```
name: "Institution 1 Node 1 server"
serverType: "node"
address: "127.0.0.1"
port: 8085
JWTPassword: "secret"
debug: true
imagesFolder: "/tank/thesis/images/institution1-node1/"
```

Slika 4.2 – Primjer konfiguracijske datoteke sustava

U konfiguracijskoj datoteci navodi se naziv poslužitelja, vrsta poslužitelja (web poslužitelj, vršni poslužitelj, poslužitelj na razini institucije ili čvor za iscrtavanje), IP adresa, port, lozinka za generiranje zaštitne sume JWT tokena, da li se pokreće u načinu za otklanjanje grešaka te direktorij gdje se spremaju DICOM zapisi. Ako je IP adresa različita od prethodne, administrator sustava je treba promijeniti, ili dodati novu ako se radi o novom poslužitelju, preko web stranice. Ovo su aktivnosti koje su potrebne za premještaj ili instalaciju sustava. Održavanje sustava nije potrebno (osim sklopovlja). Jedina intervencija koju je potrebno napraviti je zamjena lozinke u slučaju da je kompromitirana zaštita JWT tokena. Sve što je potrebno napraviti je da se u svim konfiguracijskim datotekama unese nova lozinka te da pokrenutim programima pošalje signal SIGUSR1 da ponovno učitaju konfiguracijsku datoteku. Nije potrebno ponovno pokretanje niti jednog poslužitelja. Do prekida usluge će ipak doći jer je kod korisnika učitan stari JWT token pa provjera tokena neće proći zbog nepodudaranja zaštitne sume radi toga što je zamijenjena lozinka. Korisnici se trebaju odjaviti sa sustava i ponovno prijaviti. Prekid usluge je na ovaj način puno kraći nego da se dodatno trebalo raditi i ponovno pokretanje poslužitelja.

4.2 Korisničko sučelje

Korisničko sučelje izvedeno je kao web stranica korištenjem tehnologija HTML5, CSS i JavaScript. Korišteni su razvojni okvir Angular [55] i biblioteka komponenti Material [56] tvrtke Google te programski jezik TypeScript [57] tvrtke Microsoft. Programski jezik TypeScript

je nadskup programskog jezika JavaScript s dodanim provjerama vrste podataka. Za prevođenje i pakiranje web stranice koristi se alat Angular CLI (engl. *Command Line Interface*). Angular CLI prevodi TypeScript programski kod u JavaScript te pakira i sažima rezultirajuće datoteke u nekoliko HTML, CSS i JavaScript datoteka.

Korisničko sučelje omogućava rad sa svim razinama poslužitelja kroz izbornike i ekrane specijalizirane za određene zadatke. Zadaci koji se mogu obavljati kroz korisničko sučelje su:

- Promjena lozinke
- Pregled vlastitih podataka
- Upravljanje vrstama DICOM zapisa
- Upravljanje korisnicima
- Upravljanje pacijentima
- Upravljanje poslužiteljima na razini ustanove
- Upravljanje čvorovima za iscrtavanje
- Slanje DICOM zapisa na poslužitelj
- Upravljanje DICOM zapisima
- Upravljanje radiološkim izvještajima
- Pregledavanje DICOM zapisa

Bitno je napomenuti da je nužno maksimalno sačuvati kvalitetu zapisa jer se inače mogu izgubiti sitni detalji. Radi toga korisnik prilikom pregledavanja DICOM zapisa ima mogućnost izbora između 3 oblika zapisa: PNG, JPEG i WebP [58]. Prvi oblik zapisa uvijek ima 100% kvalitete originala (najveće zauzeće slike), a druga dva su podešena tako da im je kvaliteta 95% originala, a samim tim imaju i manje zauzeće slike. Na taj način se korisniku daje izbor između kvalitete slike i brzine učitavanja. Noviji oblici zapisa kao npr. HEIF, AVIF i JPEG XL nisu podržani jer nemaju podršku od strane web preglednika ili ne postoje biblioteke koje se koriste u pozadinskoj potpori.

4.3 Pozadinska potpora

Programsko rješenje za sve četiri razine poslužitelja napravljeno je u programskom jeziku Go [59] tvrtke Google. Programski jezik Go odabran je zbog toga što ga se lagano može naučiti uz predznanje programskog jezika C i nekoliko drugih mogućnosti koje pruža.

Jedna od tih mogućnosti je automatsko oslobađanje zauzete memorije. Programer prilikom izrade programskog rješenja treba samo tražiti određenu količinu memorijskog prostora, a prevoditelj programskog jezika Go u izvršnu datoteku umetne procedure za oslobađanje zauzetog memorijskog prostora. Kod zauzeća male količine memorijskog prostora uopće nije potrebno brinuti o oslobađanju, dok kod učitavanja velikih DICOM zapisa to ne vrijedi. Procedure za oslobađanje memorijskog prostora oduzimaju korisno vrijeme rada programa te se izvršavaju s

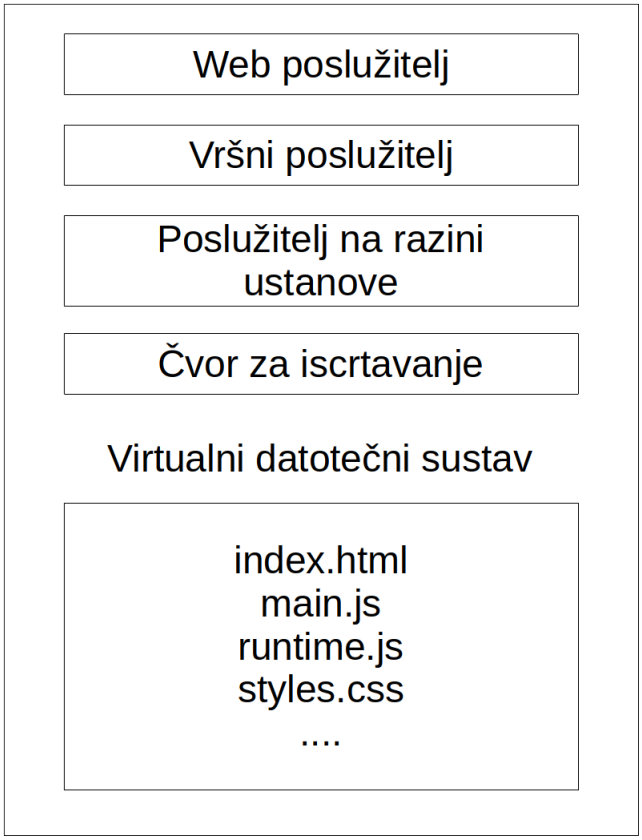
odgodom i ograničeno im je trajanje. U slučaju zauzeća velike količine memorijskog prostora do oslobađanja neće doći dugo vremena. Npr. prilikom učitavanja velikih DICOM zapisa moguće je zauzeće od nekoliko desetaka GB memorijskog prostora. U tom slučaju koristi se eksplicitan poziv procedure koja oslobađa zauzeti memorijski prostor te se omogućava učitavanje sljedećih DICOM zapisa.

Druga prednost programskog jezika Go je ta da u izvršnu datoteku zapakira sav potrebni programski kod kao i sve biblioteke koje se koriste. Klasično rješenje koje se koristi u drugim programskim jezicima je dinamičko povezivanje izvršnog koda s biblioteka. Prevoditelj programskog jezika Go nakon prevođenja programa, obilazi sve biblioteke koje se koriste te ih umeće u istu datoteku s prevedenim programskim kodom. Kao rezultat dobiva se samo jedna datoteka velika nekoliko MB ili nekoliko desetaka MB. Iako je takva datoteka velika, tako se izbjegavaju situacije da korisnik nema instalirane potrebne biblioteke ili da su instalirane biblioteke krive verzije zbog toga što ih koristi neko drugo programsko rješenje. U tom slučaju nastaju konflikti koje nije lagano riješiti jer svako programsko rješenje zahtjeva svoju verziju biblioteka, a taj konflikt nije moguće riješiti bez ažuriranja i ponovnog prevođenja nekog od programa što uglavnom nije moguće zbog nedostatka izvornog koda. Osim izvršnog koda i biblioteka u izvršnu datoteku moguće je zapisati i neke druge datoteke. Njima se pristupa kao normalnim datotekama unutar datotečnog sustava jer Go prevoditelj prilikom unošenja vanjskih datoteka stvara virtualni datotečni sustav. Ova mogućnost iskorištena je za pakiranje web stranice. Tako su sve četiri razine poslužitelja, kao i web stranica zapakirani u jednu izvršnu datoteku te nije potrebno instalirati niti održavati nezavisni web poslužitelj što korisniku olakšava posao prilikom instalacije i kasnije smanjuje potrebu za održavanjem.

Programski jezik Go ima još nekoliko prednosti od kojih će biti spomenuta samo jedna, a to je detekcija grešaka. Go prevoditelj sve nekorištene reference na vanjske biblioteke smatra greškama i zaustavlja prevođenje kada naiđe na takve slučajeve. Isto vrijedi i za nekorištene deklarirane varijable. Go prevoditelj ima ugrađen alat „vet” koji radi statičku analizu koda te uočava greške i upozorava programera. Prilikom profiliranja programa ima mogućnost detektiranja utrke između dretvi što je inače teško za otkriti i popraviti. Osim toga ima i alat za formatiranje programskog koda te je programski kod uvijek uredan i lak za čitanje i održavanje.

Dodatno su korištene sljedeće biblioteke:

- Gin [60] - za izvođenje REST poslužitelja
- GORM [61] - biblioteka za preslikavanje programskih objekata u SQL upite
- glfw, gl i mgl32 [62] - biblioteke za upravljanje OpenGL kontekstom, pristup OpenGL-u i linearnu algebru
- dicom [63] - biblioteka za učitavanje DICOM zapisa
- go-webp [64] - biblioteka za učitavanje i spremanje slika u WebP obliku
- quic-go [65] - biblioteka za HTTP/3 protokol

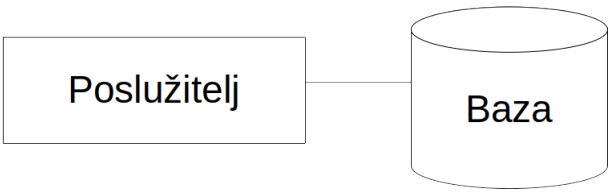


Slika 4.3 – Zapakirana izvršna datoteka

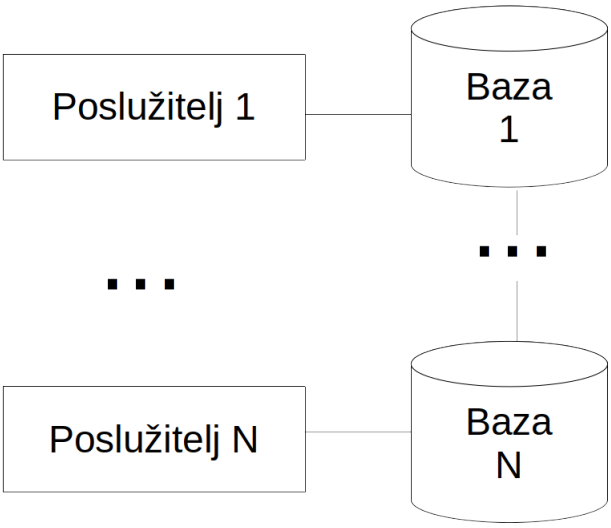
- pq [66] - biblioteka za spajanje na Postgres bazu podataka
- go-yaml [67] - podrška za rad sa YAML zapisima
- golang-jwt [68] i gin-jwt [69] - biblioteke za rad s JWT tokenima

4.4 Baza podataka

Za upravljanje bazama podataka koristi se CockroachDB [70]. To je pogon baze podataka (engl. *database engine*) koji za upite koristi jezik SQL kompatibilan s Postgres bazom podataka. Za pohranu podataka koristi NoSQL bazu podataka temeljenu na zapisima ključ-vrijednost (engl. *key-value*). Prednost NoSQL nad SQL bazama podataka je lagana distribucija i replikacija. CockroachDB koristi najbolje karakteristike SQL i NoSQL baza podatka i na taj način se mogu izrađivati distribuirane baze podataka te formirati grozdove. Kako model i arhitektura sustava ne dopuštaju direktnu komunikaciju između poslužitelja, moguće je podići nekoliki instanci iste vrste poslužitelja, a njihove baze podataka povezati u grozd. Dopuštene veze između poslužitelja i baza podataka su 1:1, 1:N i M:N. Preporučene konfiguracije su takve da se na istom računalu pokreću poslužitelj i baza podataka. U slučaju potrebe, moguće je pokrenuti još jedno ili više računala, svako sa svojim poslužiteljem i bazom podataka.



Slika 4.4 – Jedan poslužitelj sa jednom bazom podataka



Slika 4.5 – N poslužitelja sa N baza podataka

Poglavlje 5

Kvalitativna i kvantitativna tehnička analiza i vrednovanje svojstvenih performansi izgrađenog programskog sustava

U ovom poglavlju prikazani su rezultati testiranja izgrađenog programskog sustava. Testiranja su provedena na DICOM zapisima s javno dostupnih repozitorija [71] [72] [73].

Računalna konfiguracija na kojoj su provedana testiranja:

- Centralni procesor: Intel i7-11700, osnovni takt 2.5 GHz, maksimalni takt 4.9 GHz, 8C16T
- Grafički procesor: Intel UHD Graphics 750, integriran s centralnim procesorom
- Radna memorija: 64 GB RAM, DDR4-3200, dvokanalna
- Tvrdi disk: Samsung SSD 970 EVO, 1TB, NVMe
- Datotečni sustav: OpenZFS 2.0.2
- Web preglednik: Chromium 93

Zapisi su odabrani tako da zauzeće bude u području od oko 1 MB do 1GB te da je svaki slijedeći otprilike za red veličine veći od prethodnog (tablica 5.1).

Testiranja su provedena tako da je pokrenut sustav s jednim web poslužiteljem, jednim vršnim poslužiteljem, jednim poslužiteljem na razini ustanove te jednim čvorom za iscrtavanje. Zapisi su pregledavani na istom računalu na tri načina: bez simulacije mreže, sa simulacijom 4G mreže i sa simulacijom 3G mreže.

Izračunato je zauzeće ulazne 2D teksture i 3D teksture za pohranu. Nakon prikaza pojedine slike, iz dimenzija ekrana izračunato je zauzeće pojedine izlazne 2D teksture. Za iscrtavanje slojeva i volumena korištena je izlazna tekstura koja sadrži samo jedan kanal boje. Za iscrtavanje volumena korišten je jednoprolazni program za sjenčanje elemenata slike temeljen na odašiljanju zrake i ograničenjem od 100 koraka. To ograničenje odabrano je radi toga da uvjeti testiranja

budu jednaki za sve slike iako bi u praktičnoj primjeni broj koraka ovisio o dimenzijama slike.

Oznaka	Širina	Visina	Broj slojeva	Zauzeće [B]
A	512	512	1	526.554
B	512	512	10	5.250.756
C	512	512	54	28.333.048
D	512	512	555	291.295.842
E	1996	2457	96	941.643.712

Tablica 5.1 – Podaci o testnim DICOM zapisima

Oznaka	Širina	Visina	Zauzeće [B]
A	512	512	524.288
B	512	512	524.288
C	512	512	524.288
D	512	512	524.288
E	1996	2457	9.808.344

Tablica 5.2 – Zauzeće memorije ulazne 2D teksture

5.1 Učitavanje DICOM zapisa sa tvrdog diska u 3D teksturu

Prema specifikaciji proizvođača tvrdi disk postiže brzinu prijenosa podataka od 3500 MB/s što znači da se zapis veličine 1 GB učitava u manje od $\frac{1}{3}$ sekunde. U slučaju slike E iz tablice 5.3 vidimo da učitavanje traje oko 11 s. Može se zaključiti da učitavanje snimke ne ovisi o brzini tvrdog diska nego o brzini dekomprimiranja sadržaja DICOM zapisa i o učitavanju u OpenGL teksture. Ovdje dolazi do izražaja nedostatak OpenGL, a to je da se izvodi na jednoj dretvi u upravljačkom programu sklopovlja.

Za pohranu podataka korišten je datotečni sustav ZFS sa sljedećim postavkama:

- bez komprimiranja i kriptiranja
- s komprimiranjem bez kriptiranja
- bez komprimiranja s kriptiranjem
- s komprimiranjem i kriptiranjem

Korištenjem komprimiranja nije došlo do smanjenja veličine DICOM zapisa što je i očekivano jer se za pohranu slojeva koriste algoritmi za komprimiranje slika koji su vrlo učinkoviti.

Po rezultatima prikazanim u tablici 5.3 vidi se da su razlike u sva četiri slučaja vrlo male. Stoga se preporučuje korištenje ZFS datotečnog sustava s kriptiranjem podataka jer nema gubitka performansi, a povećava se sigurnost podataka.

Oznaka	Nekriptirano		Kriptirano	
	Nekomprimirano [s]	Komprimirano [s]	Nekomprimirano [s]	Komprimirano [s]
A	0,014	0,015	0,015	0,015
B	0,091	0,087	0,084	0,086
C	0,437	0,433	0,438	0,432
D	3,594	3,520	3,653	3,534
E	11,178	10,961	11,080	11,253

Tablica 5.3 – Vremena učitavanja DICOM zapisa sa tvrdog diska

Oznaka	Širina	Visina	Dubina	Zauzeće [B]
A	512	512	1	524.288
B	512	512	10	5.242.880
C	512	512	54	14.155.776
D	512	512	555	145.489.920
E	1664	2048	96	327.155.712

Tablica 5.4 – Zauzeće memorije 3D teksture za pohranu

5.2 Udaljeno iscrtavanje i prijenos bez simulacije mreže

Kako se iscrtavanje i prikaz slike obavljaju na istom računalu vrijeme prijenosa je vrlo kratko (ispod 1 ms) pa ga zanemarujemo. Na primjeru slika B, C i D (tablica 5.5) koje su istih dimenzija možemo zaključiti da vrijeme iscrtavanja ovisi o popunjenosti slike, odnosno algoritmu za sažimanje.

U tablici 5.6 prikazana su vremena iscrtavanja i prijenosa volumena. Dok su veličine slika u pravilu manje, vremena iscrtavanja značajno su veća. Ovo je očekivano zbog toga što je iscrtavanje volumena računski puno zahtjevnije u odnosu na iscrtavanje slojeva.

Oznaka	PNG		JPEG		WebP	
	Vrijeme [ms]	Veličina [kB]	Vrijeme [ms]	Veličina [kB]	Vrijeme [ms]	Veličina [kB]
A	69,33	222,0	23,00	116	62,00	74,9
B	17,67	69,4	5,67	45,8	39,00	21,6
C	16,00	142	7,33	95	48,33	64,6
D	14,00	86,2	6,33	46,6	42,33	30,3
E	32,00	83,1	6,00	39,4	36,33	19,4

Tablica 5.5 – Vremena iscrtavanja sloja i prijenosa bez simulacije mreže

Oznaka	PNG		JPEG		WebP	
	Vrijeme [ms]	Veličina [kB]	Vrijeme [ms]	Veličina [kB]	Vrijeme [ms]	Veličina [kB]
A	-	-	-	-	-	-
B	42,00	51,4	14,67	26,8	55,67	7,7
C	100,00	107	16,00	51,1	57,67	23
D	61,33	68,3	16,00	32,7	56,67	11,4
E	99,00	82,8	27,33	35,1	61,33	9,9

Tablica 5.6 – Vremena iscrtavanja volumena i prijenosa bez simulacije mreže

Oznaka	Širina	Visina	Zauzeće [B]
A	736	736	541.696
B	688	688	473.344
C	688	688	473.344
D	688	688	473.344
E	559	688	384.592

Tablica 5.7 – Zauzeće memorije izlazne 2D teksture

5.3 Udaljeno iscrtavanje i prijenos preko simulirane 4G mreže

Simulirani su sljedeći uvjeti mreže:

- •Brzina primanja: 4.0 Mb/s
- •Brzina slanja: 3.0 Mb/s
- •Latencija: 50 ms

U ovom primjeru vrijeme prijenosa više nije zanemarivo i veličina slike sada ima utjecaj na vrijeme prijenosa. Najbolje vrijeme ima WebP algoritam iako je po rezultatima iz tablice 5.5

najsporiji. Razlog je taj što je sporost algoritma prikrivena latencijom mreže i brzinom prijenosa, a kako algoritam daje slike najmanje veličine trajanje prijenosa je najkraće.

Oznaka	PNG	JPEG	WebP
	Vrijeme [ms]	Vrijeme [ms]	Vrijeme [ms]
A	800,33	285,67	217,67
B	195,00	146,33	98,00
C	351,00	247,00	185,00
D	226,00	168,00	116,33
E	222,00	133,33	91,33

Tablica 5.8 – Vremena iscrtavanja sloja i prijenosa na simuliranoj 4G mreži

Oznaka	PNG	JPEG	WebP
	Vrijeme [ms]	Vrijeme [ms]	Vrijeme [ms]
A	-	-	-
B	157,00	108,00	70,00
C	328,33	156,00	113,33
D	201,00	118,67	78,00
E	272,00	125,67	84,00

Tablica 5.9 – Vremena iscrtavanja volumena i prijenosa na simuliranoj 4G mreži

5.4 Udaljeno iscrtavanje i prijenos preko simulirane 3G mreže

Simulirani su sljedeći uvjeti mreže:

- Brzina primanja: 1.5 Mb/s
- Brzina slanja: 750 kb/s
- Latencija: 100 ms

Dobiveni rezultati dodatno potvrđuju zaključak iz poglavlja 5.3 te pokazuju da algoritam WebP daje uvjerljivo najbolje rezultate radi toga što je utjecaj latencije mreže i brzine prijenosa dominantan u odnosu na vrijeme sažimanja.

Oznaka	PNG	JPEG	WebP
	Vrijeme [ms]	Vrijeme [ms]	Vrijeme [ms]
A	3193,33	1322,33	505,33
B	607,67	351,00	219,67
C	1676,67	1156,67	737,33
D	893,67	603,00	272,33
E	624,67	316,33	209,67

Tablica 5.10 – Vremena iscrtavanja sloja i prijenosa na simuliranoj 3G mreži

Oznaka	PNG	JPEG	WebP
	Vrijeme [ms]	Vrijeme [ms]	Vrijeme [ms]
A	-	-	-
B	404,33	243,33	158,00
C	1323,33	523,33	229,00
D	581,67	282,00	164,00
E	760,00	310,67	161,67

Tablica 5.11 – Vremena iscrtavanja volumena i prijenosa na simuliranoj 3G mreži

5.5 Sažetak rezultata testiranja

Iz rezultata testiranja može se zaključiti da je vrijeme iscrtavanja u slučajevima stvarnih uvjeta na mreži prikriveno latencijom i propusnošću mreže. Trajanje prijenosa ovisi o veličini slike, a veličina o korištenom algoritmu za sažimanje slike. Najbolje rezultate dao je algoritam WebP. Noviji algoritmi za sažimanje slika, kao npr. HEIF [74], AVIF [75] i JPEG XL [76], nisu testirani jer u trenutku pisanja rada nisu podržani na web preglednicima ili ne postoje biblioteke za programski jezik Go.

Poglavlje 6

Zaključak

6.1 Glavni zaključci i doprinosi

Za prikazivanje DICOM zapisa lokalno na računalu korisnika, kao i unutar ustanove, postoji velik broj specijaliziranih rješenja. Također postoje rješenja, iako u manjem broju, za daljinsko iscrtavanje DICOM zapisa. Ovim istraživanjem željelo se napraviti rješenje koje može raditi na razini od države, preko ustanove do računala korisnika. Pritom se vodilo računa da količina prenesenih podataka bude što manja te da se postojeće sklopovlje iskoristi što više. Značajan dio istraživanja posvećen je sigurnosti prijenosa podataka preko mreže, kao i lokalnoj pohrani podataka. Napravljeno je programsko rješenje koje korisniku omogućava lako korištenje sustava bez potrebe poznavanja arhitekture sustava. Kombinacijom različitih postupaka asinkronog programiranja osigurano je smanjenje latencije prijenosa podataka tako da korisnik ima osjećaj trenutnog odziva sustava. Programsko rješenje napravljeno je tako da se može kopirati i pokrenuti bez potrebe instaliranja, a održavanje sustava svedeno je na minimum. Moguće je korištenje starijeg sklopovlja tako da korisnici ne moraju nužno investirati u novu opremu, što u trenutku pisanja ovog rada nije niti moguće zbog nestašice poluvodiča na tržištu. Indirektno se doprinijelo zaštiti okoliša zbog toga što više nema potrebe za prijenosom DICOM zapisa na optičkim medijima. Osim toga, smanjen je bespotreban kontakt liječnika i pacijenata u svrhu prijenosa optičkog medija što je osobito bitno s obzirom na COVID-19 pandemiju.

Prema mišljenju autora, glavni znanstveni doprinosi ove disertacije su sljedeći:

- Izrađen je model učinkovite programske potpore i procedure za višekorisničku udaljenu vizualizaciju volumnih medicinskih podataka
- Napravljena je platformski neovisna arhitektura programske potpore zasnovana na vizualizacijsko-komunikacijskom radnom toku
- Provedena je kvalitativna i kvantitativna tehnička analiza i vrednovanje svojstvenih performansi izgrađenog programskog sustava

6.2 Daljnja istraživanja

Izrađeno programsko rješenje omogućava korisnicima sustava dostupnost podataka u kratkom vremenu nakon izrade snimke, bez obzira na fizičku udaljenost. Međutim, programsko rješenje je manjkavo u funkcionalnostima koje korisnici trebaju u svakodnevnom radu, a to je mjerenje dimenzija na zapisima. Zbog zadanih okvira istraživanja ovaj dio je izostavljen jer nije ostvariv bez korištenja naprednih funkcionalnost HTML5 norme. Radi se o normi WebGL [77] koja je temeljena na OpenGL ES-u [78] (podskup OpenGL optimiran za rad na ugrađenim sustavima), ali se izvodi unutar web preglednika. Prethodno je potrebno istražiti koliki udio računala potencijalnih korisnika ima mogućnosti korištenja WebGL-a s obzirom na to da se većinom radi o pod-optimalnim računalima.

Sljedeće istraživanje moglo bi obuhvatiti dinamičku prilagodbu svjetline umjesto korištenja centra i širine prozora definiranih u DICOM zapisu. Moguća su dva pristupa: omogućiti korisniku da sam definira centra i širinu prozora ili ih automatski definirati iz sadržaja elemenata slike. Na ovaj način bi se korisniku omogućilo ili olakšalo uočavanje detalja koji možda nisu vidljivi na osnovu predefiniranih postavki.

Buduće istraživanje moglo bi istražiti alternative OpenGL-u. S obzirom na to da je OpenGL temeljen na složenom automatu stanja i izvodi se na jednoj dretvi unutar upravljačkog programa, ne iskorištava dobro moderne višejezgrene procesore. Kao zamjenu za OpenGL može se istražiti Vulkan [79] aplikacijsko programsko sučelje. Vulkan je također namijenjen za rad s računalnom grafikom, ali nije temeljen na automatu stanja nego je izveden kao tanak sloj između korisničkog programa i upravljačkog programa. Zbog toga je podržano višedretveno izvođenje, a složeno upravljanje stanjima prebačeno je na korisnički program gdje je to lakše izvedivo nego unutar upravljačkog programa grafičkog sklopovlja. Osim grafičkih programskih sučelja može se istražiti i računsko programsko sučelje opće namjene OpenCL [80]. Način korištenja OpenCL-a sličan je načinu korištenja OpenGL-a pa se programska rješenja mogu preraditi tako da ga podržavaju. OpenCL se za razliku od OpenGL-a može koristiti i na centralnim procesorima, a ne samo na specijaliziranom grafičkom sklopovlju. Na ovaj način bi se omogućilo korištenje poslužitelja koji nemaju grafičko sklopovlje i ne mogu poslužiti kao čvor za iscrtavanje. Osim toga, moguće je istovremeno iskoristiti i centralni procesor i specijalizirano grafičko sklopovlje te tako povećati iskorištenost poslužitelja.

Dodatak A

Primjer specifikacije komunikacijskog sučelja u YAML obliku

```yaml
openapi: 3.0.3
servers:
  - url: "https://localhost:8080/restful"
info:
  title: Node server API
  description: RESTful API for node server
  version: 1.0.0
  contact:
    name: Krešimir Jozić
    email: kjozic@gmail.com
  license:
    name: MIT
paths:
  /storage/save-image:
    post:
      summary: Save image
      tags:
        - storage
        - regular_access
      description: |
        The /storage/save-image endpoint is used to save image.
      security:
        - JWTToken: []
      responses:
```

```
        "200":
          description: File checksum (used as a file name).
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/imageInfo"
        "400":
          $ref: "#/components/responses/ErrorParsingRequest"
        "401":
          $ref: "#/components/responses/ErrorUnauthorized"
        "500":
          $ref: "#/components/responses/ErrorDeletingRecord"
      requestBody:
        content:
          multipart/form-data:
            schema:
              $ref: "#/components/schemas/fileUpload"
components:
  schemas:
    imageInfo:
      type: object
      required:
        - md5sum
        - rows
        - cols
        - frames
      properties:
        md5sum:
          type: string
          description: MD5 encoded file content.
          minLength: 32
          maxLength: 32
          example: C4CA4238A0B923820DCC509A6F75849B
        rows:
          type: integer
          format: int32
          description: Number of rows
```

```
        example: 1024
      cols:
        type: integer
        format: int32
        description: Number of cols
        example: 1024
      frames:
        type: integer
        format: int32
        description: Number of frames
        example: 1024
      rescale_slope:
        type: number
        description: Rescale slope
        example: 1.0
      rescale_intercept:
        type: number
        description: Rescale intercept
        example: 0.0
      window_center:
        type: number
        description: Window center
        example: 0.0
      window_width:
        type: number
        description: Rescale slope
        example: 1.0
  fileUpload:
    type: object
    required:
      - file
    properties:
      file:
        type: string
        format: binary
        description: File content
        example: aB$13c!
```

```
  securitySchemes:
    JWTToken:
      type: http
      description: JWT token
      scheme: bearer
      bearerFormat: JWT
  responses:
    ErrorUnauthorized:
      description: User is not authorized for this action.
    ErrorQueryingDatabase:
      description: Error while querying database.
    ErrorParsingRequest:
      description: Error while parsing request data.
    ErrorDeletingRecord:
      description: Error while deleting record.
    ErrorNotFound:
      description: Endpoint not found.
tags:
  - name: info
  - name: storage
  - name: render
  - name: admin_access
    description: Path can be accessed only by administrator.
  - name: regular_access
    description: Path can be accessed by regular users.
```

Dodatak B

Primjer metapodataka o DICOM zapisu

```
# Dicom-Meta-Information-Header
# Used TransferSyntax: Little Endian Explicit
(0002,0000) UL 72                                    #   4, 1 FileMetaInformationGroupLength
(0002,0001) OB 00\01                                 #   2, 1 FileMetaInformationVersion
(0002,0010) UI =LittleEndianExplicit                 #  20, 1 TransferSyntaxUID
(0002,0013) SH [GIMP Dicom Plugin 1.0]               #  22, 1 ImplementationVersionName

# Dicom-Data-Set
# Used TransferSyntax: Little Endian Explicit
(0008,0000) UL 182                                   #   4, 1 GenericGroupLength
(0008,0008) CS [ORIGINAL\PRIMARY]                    #  16, 2 ImageType
(0008,0016) UI =SecondaryCaptureImageStorage         #  26, 1 SOPClassUID
(0008,0020) DA [20210903]                            #   8, 1 StudyDate
```

```
(0008,0021) DA [20210903]                    #   8, 1 SeriesDate
(0008,0022) DA [20210903]                    #   8, 1 AcquisitionDate
(0008,0023) DA [20210903]                    #   8, 1 ContentDate
(0008,0030) TM [000000.000000]               #  14, 1 StudyTime
(0008,0050) SH (no value available)          #   0, 0 AccessionNumber
(0008,0060) CS [MR]                          #   2, 1 Modality
(0008,0064) CS [WSD]                         #   4, 1 ConversionType
(0008,0090) PN (no value available)          #   0, 0 ReferringPhysicianName
(0010,0000) UL 60                            #   4, 1 GenericGroupLength
(0010,0010) PN [DOE^WILBER]                  #  10, 1 PatientName
(0010,0020) LO [314159265]                   #  10, 1 PatientID
(0010,0030) DA [20210903]                    #   8, 1 PatientBirthDate
(0010,0040) CS (no value available)          #   0, 0 PatientSex
(0020,0000) UL 40                            #   4, 1 GenericGroupLength
(0020,0010) IS [1]                           #   2, 1 StudyID
(0020,0011) IS [1]                           #   2, 1 SeriesNumber
(0020,0012) IS [1]                           #   2, 1 AcquisitionNumber
(0020,0013) IS [1]                           #   2, 1 InstanceNumber
(0028,0000) UL 92                            #   4, 1 GenericGroupLength
(0028,0002) US 3                             #   2, 1 SamplesPerPixel
(0028,0004) CS [RGB]                         #   4, 1 PhotometricInterpretation
(0028,0006) US 0                             #   2, 1 PlanarConfiguration
(0028,0010) US 512                           #   2, 1 Rows
(0028,0011) US 512                           #   2, 1 Columns
```

```
(0028,0100) US 8                                    #   2, 1 BitsAllocated
(0028,0101) US 8                                    #   2, 1 BitsStored
(0028,0102) US 7                                    #   2, 1 HighBit
(0028,0103) US 0                                    #   2, 1 PixelRepresentation
(7fe0,0000) UL 786444                               #   4, 1 GenericGroupLength
```

Dodatak C

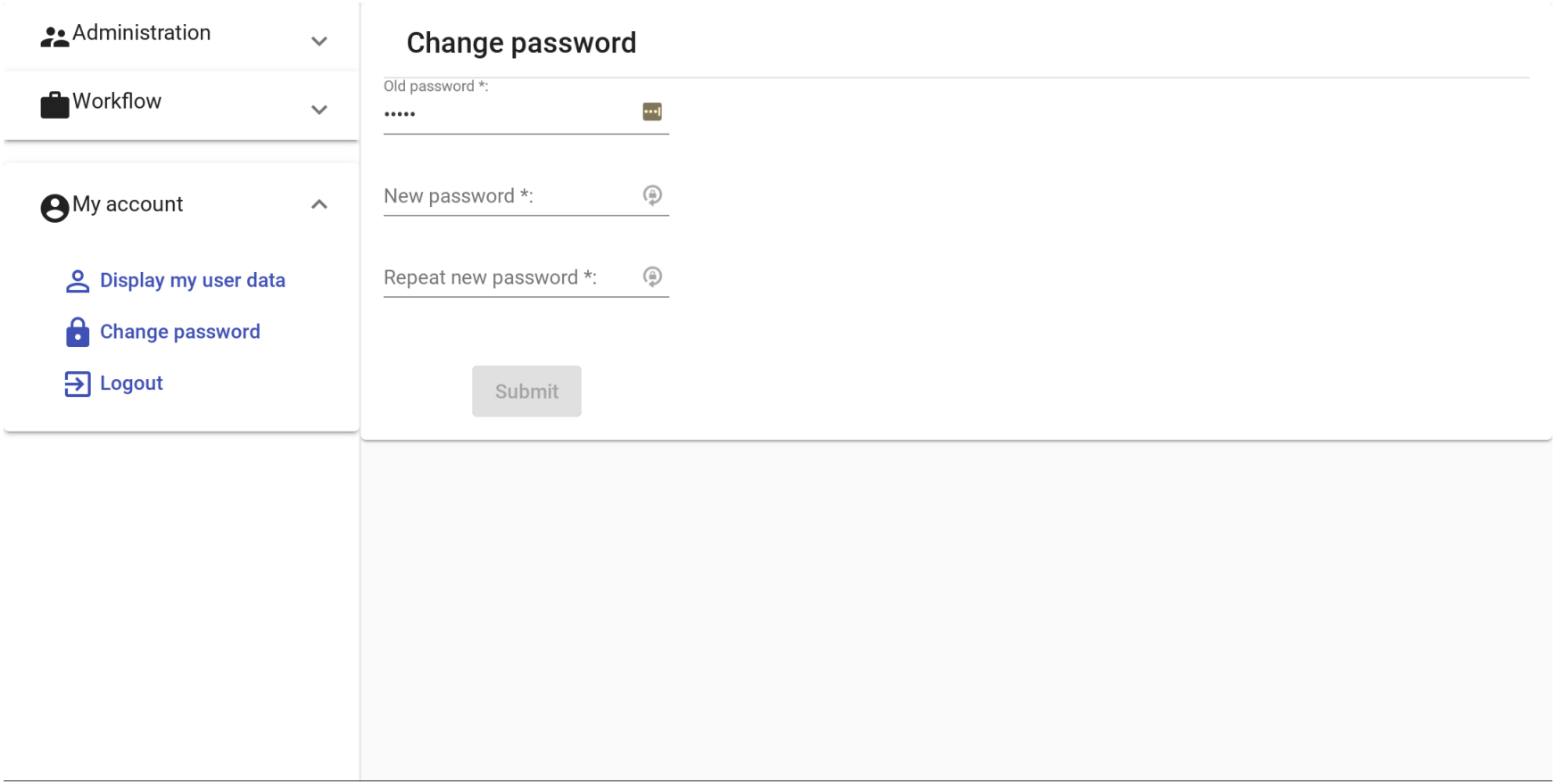
Snimke zaslona korisničkog sučelja

Zbog velikog broja ekrana prikazani su samo neki primjeri. Ekrani se mogu svrstati u nekoliko grupa.

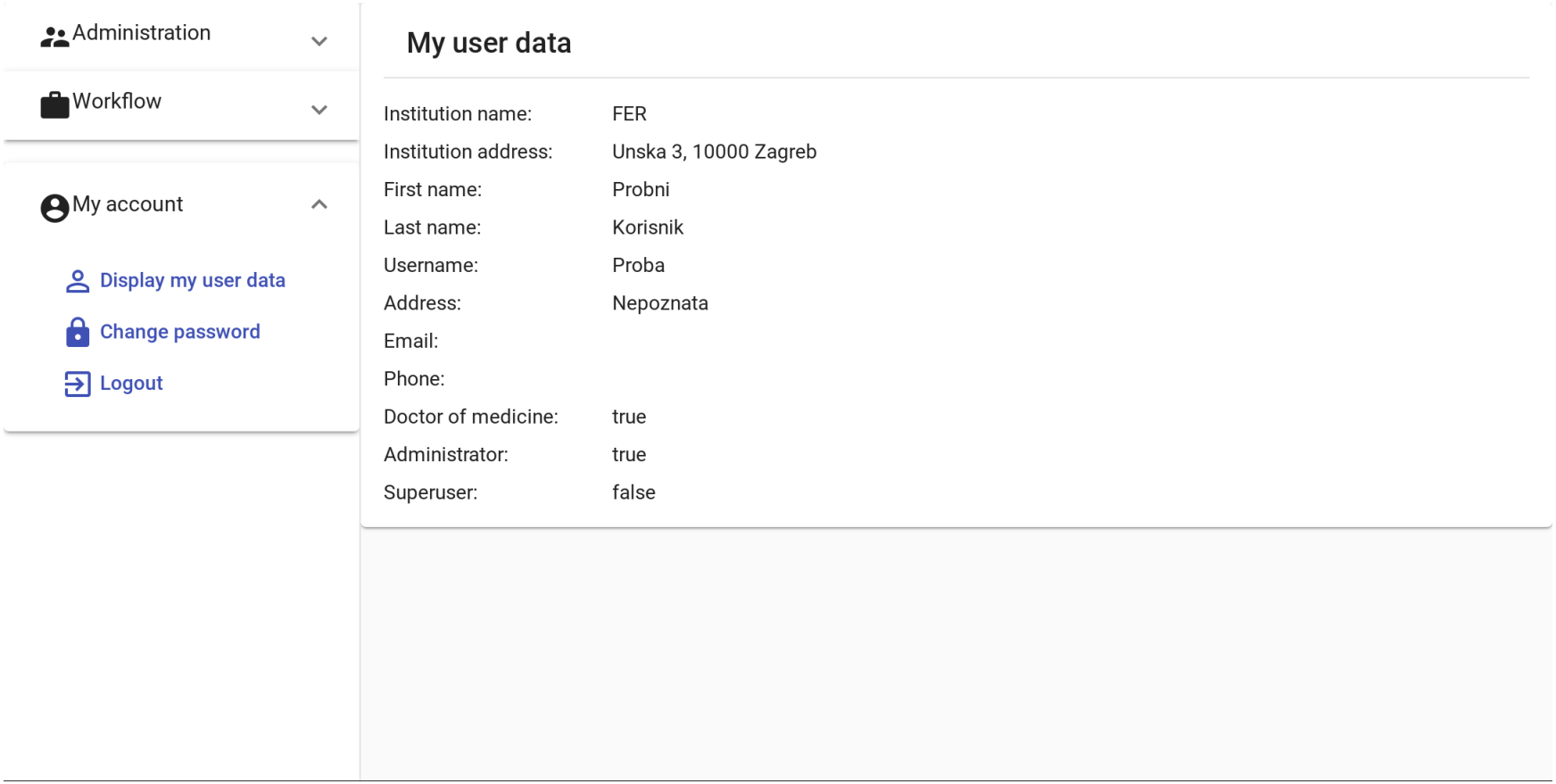
Prva grupa su jednostavni ekrani, npr. promjena lozinke C.1, prikaz vlastitih podataka C.2 ili slanje DICOM zapisa na poslužitelj C.7.

Druga grupa su ekrani za upravljanje popisima ustanova, čvorova za iscrtavanje, korisnika, pacijenata, vrstama DICOM zapisa i DICOM zapisima. Početni ekran je uvijek popis koji se može filtrirati po određenim parametrima C.3 C.4 C.6 C.8. U dnu ekrana su gumbi koji omogućavaju dodavanje novog zapisa te uređivanje C.5 C.9 i brisanje postojećeg.

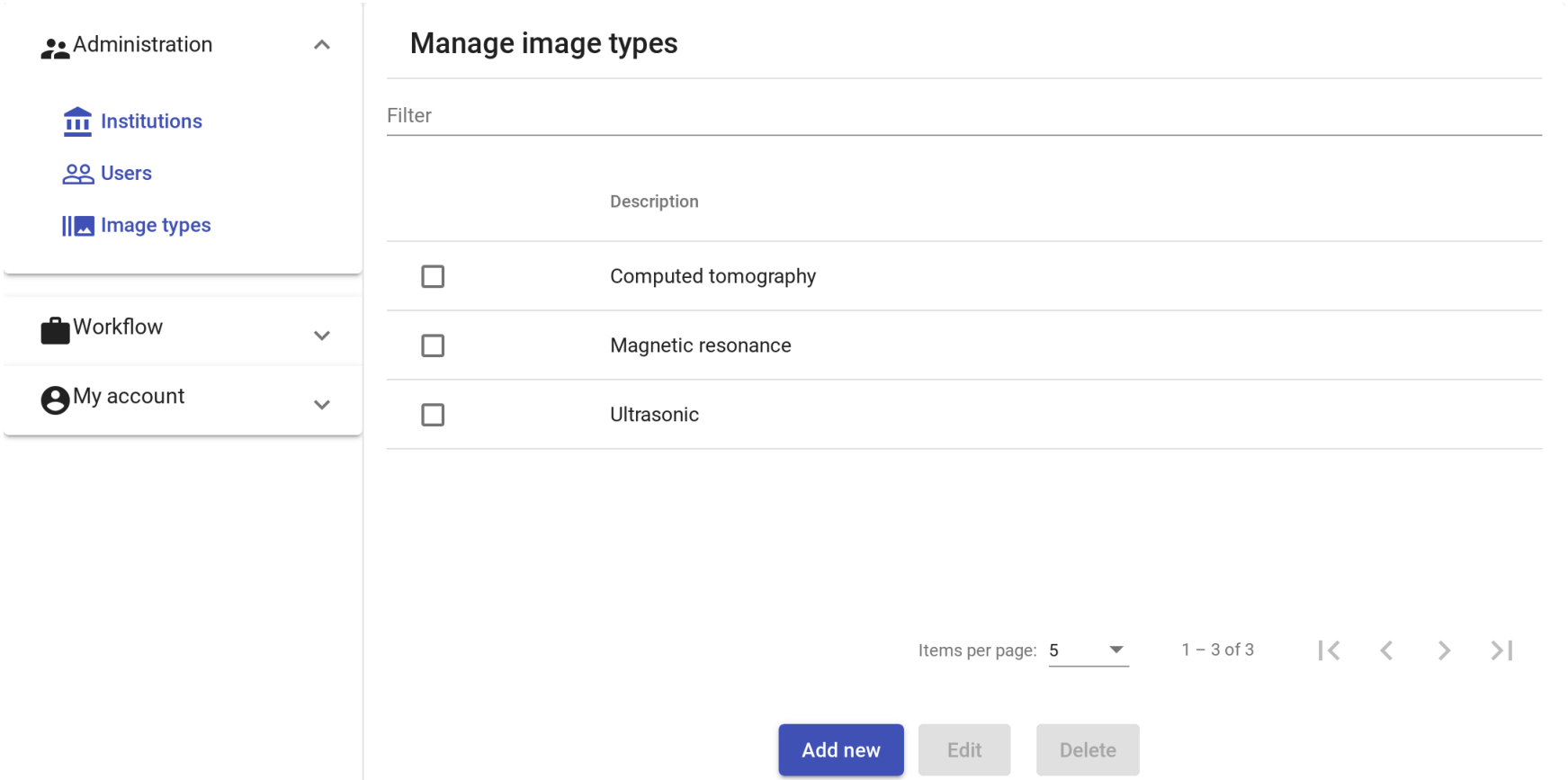
Treću grupu predstavlja ekran za izradu radiološkog izvještaja. Ekran se sastoji od nekoliko dijelova: pretraga po pacijentima, pretraga zapisa po ustanovama za odabranog pacijenta te uređivanje izvještaja C.10. U dijelu za uređivanje izvještaja moguće je uz izvještaj povezati DICOM zapise te ih pregledavati C.11. Zapise je moguće pregledavati kao slojeve slika ili kao volumne podatke. U slučaju da zapis ima samo jedan sloj, skrivaju se klizači za odabir sloja i 3D koordinate pogleda te gumb za prikaz volumena. Korisniku se pruža mogućnost da odabere kontinuirano listanje pomoću strelica ili klizača. U tom slučaju ne mora odabrati sloj pa kliknuti na gumb za dohvaćanje sloja. Prilikom kontinuiranog listanja radi se mala pauza između zadnjeg odabranog sloja i dohvata s poslužitelja. Na ovaj način se sprječava da korisnik u kratkom vremenu pošalje nekoliko zahtjeva prema poslužitelju jer se ionako prikazuje samo zadnji odabrani sloj.



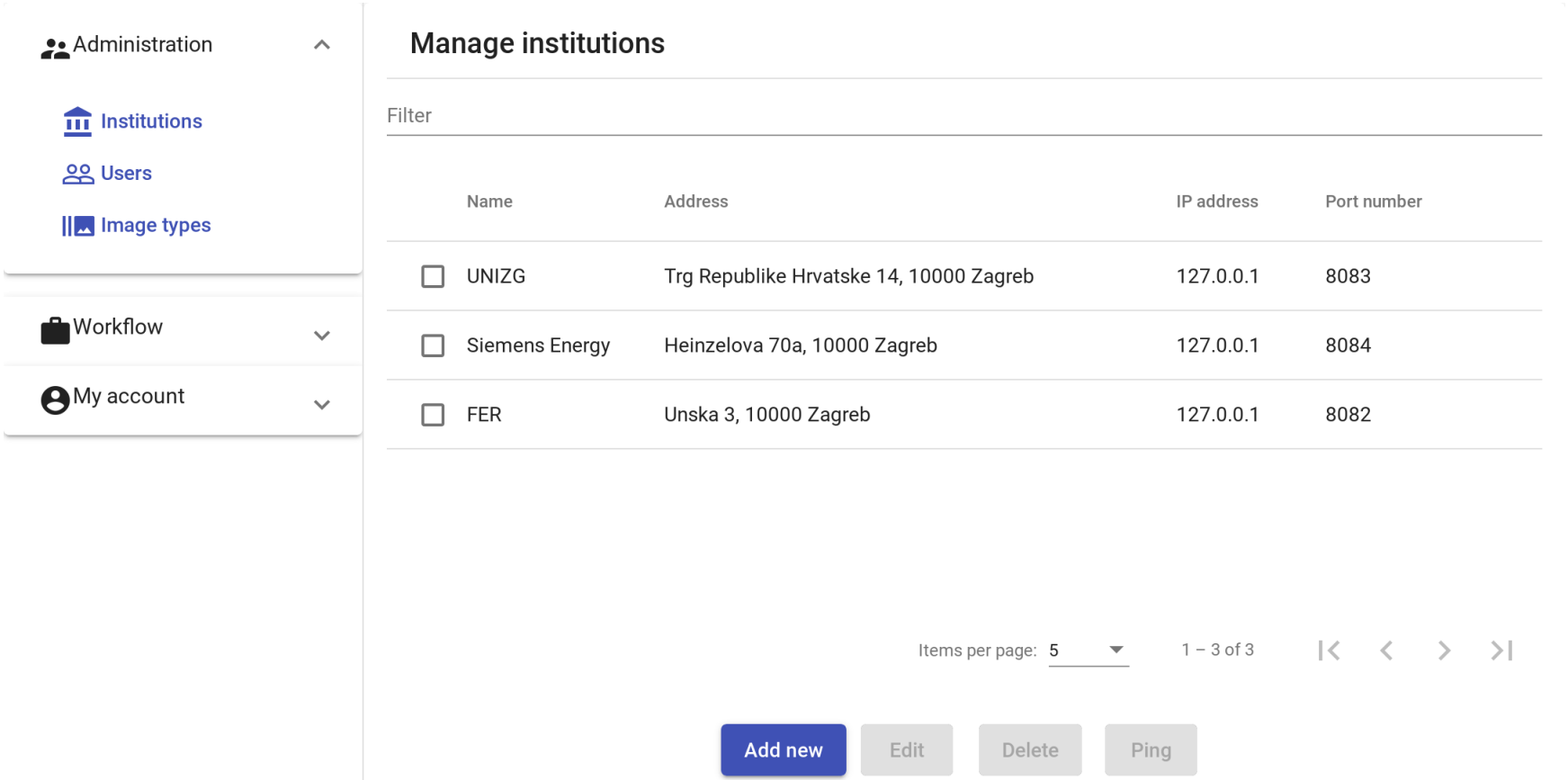
Slika C.1 – Promjena lozinke



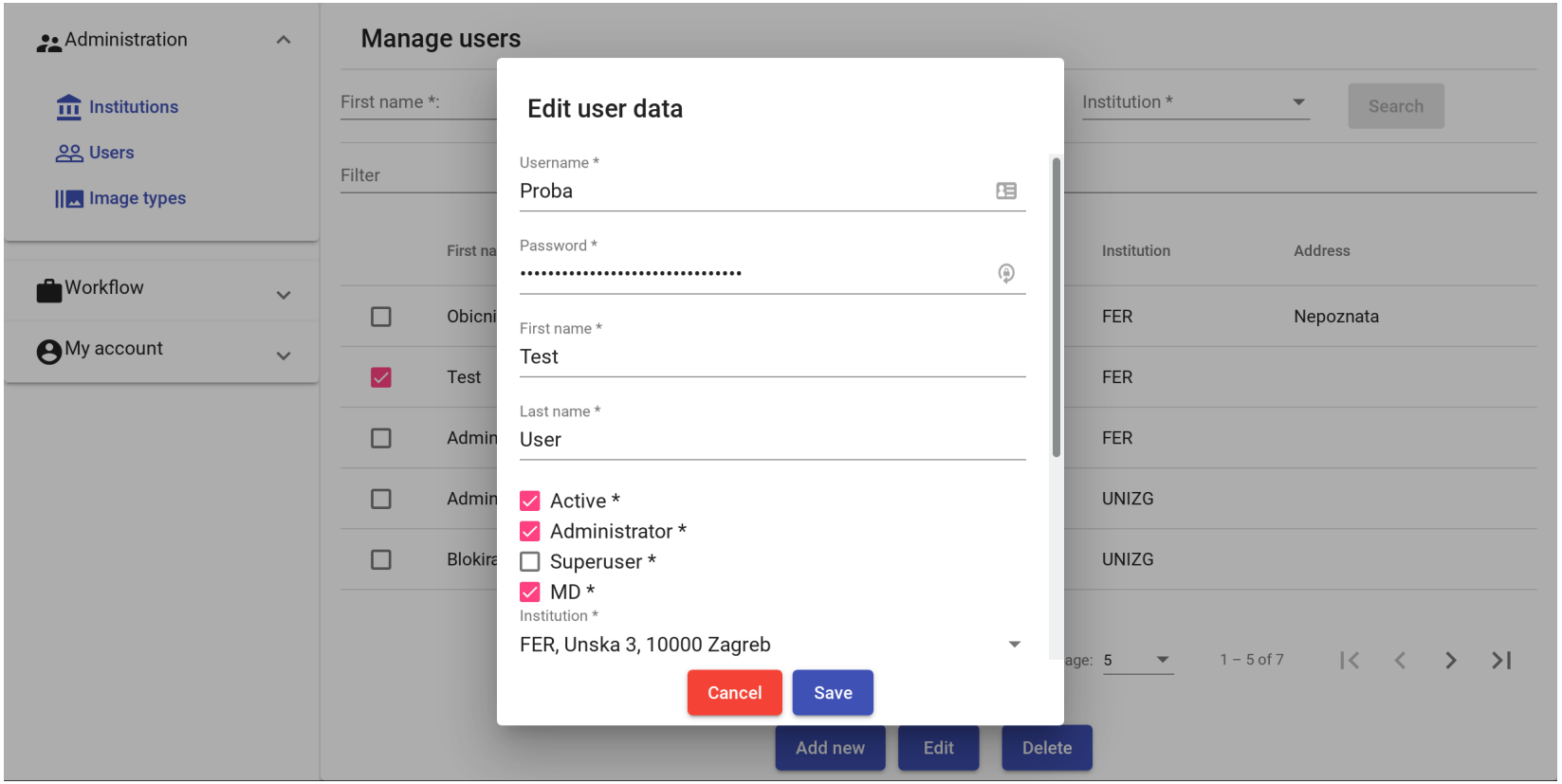
Slika C.2 – Prikaz vlastitih podataka



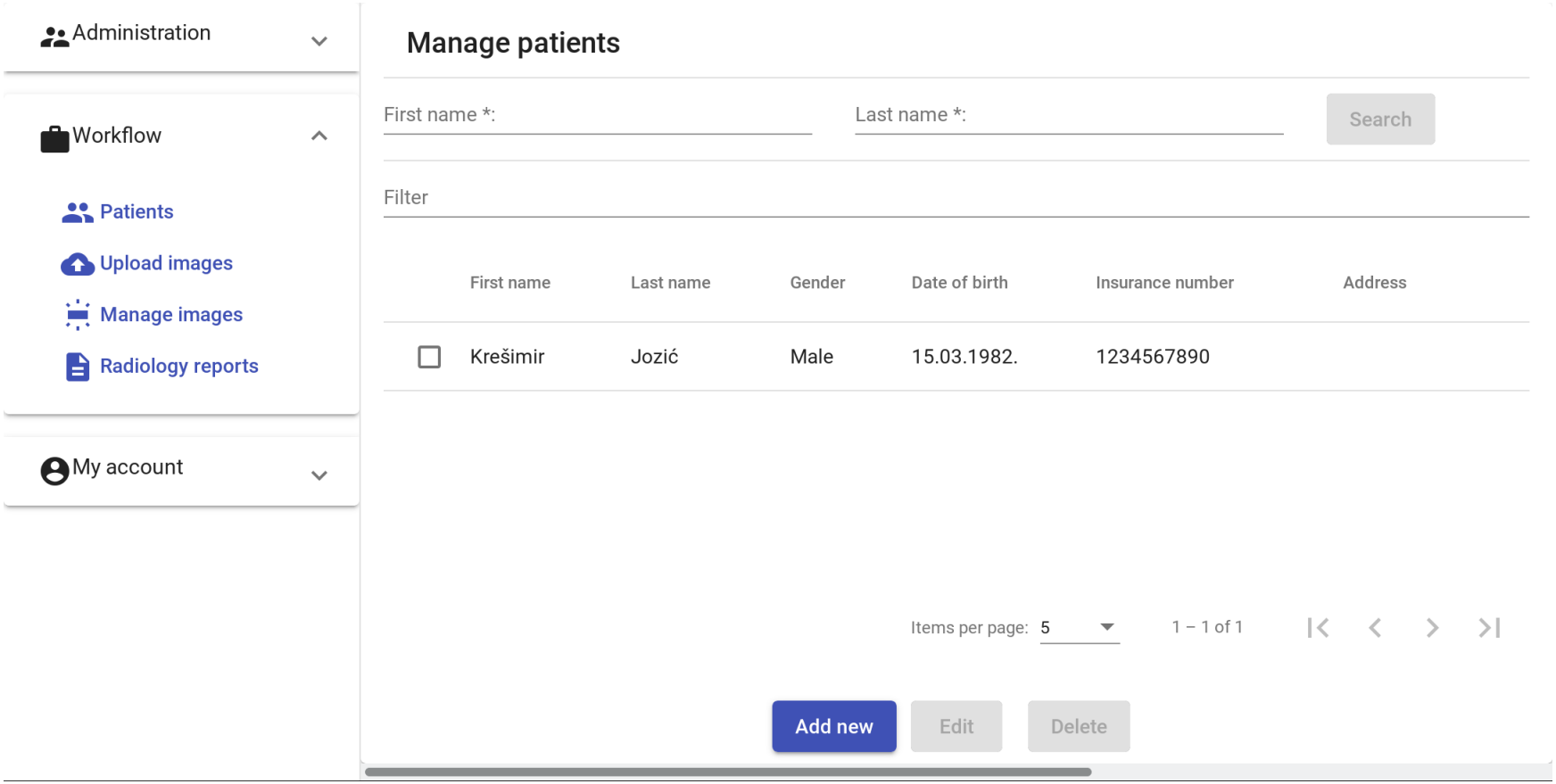
Slika C.3 – Upravljanje vrstama DICOM zapisa



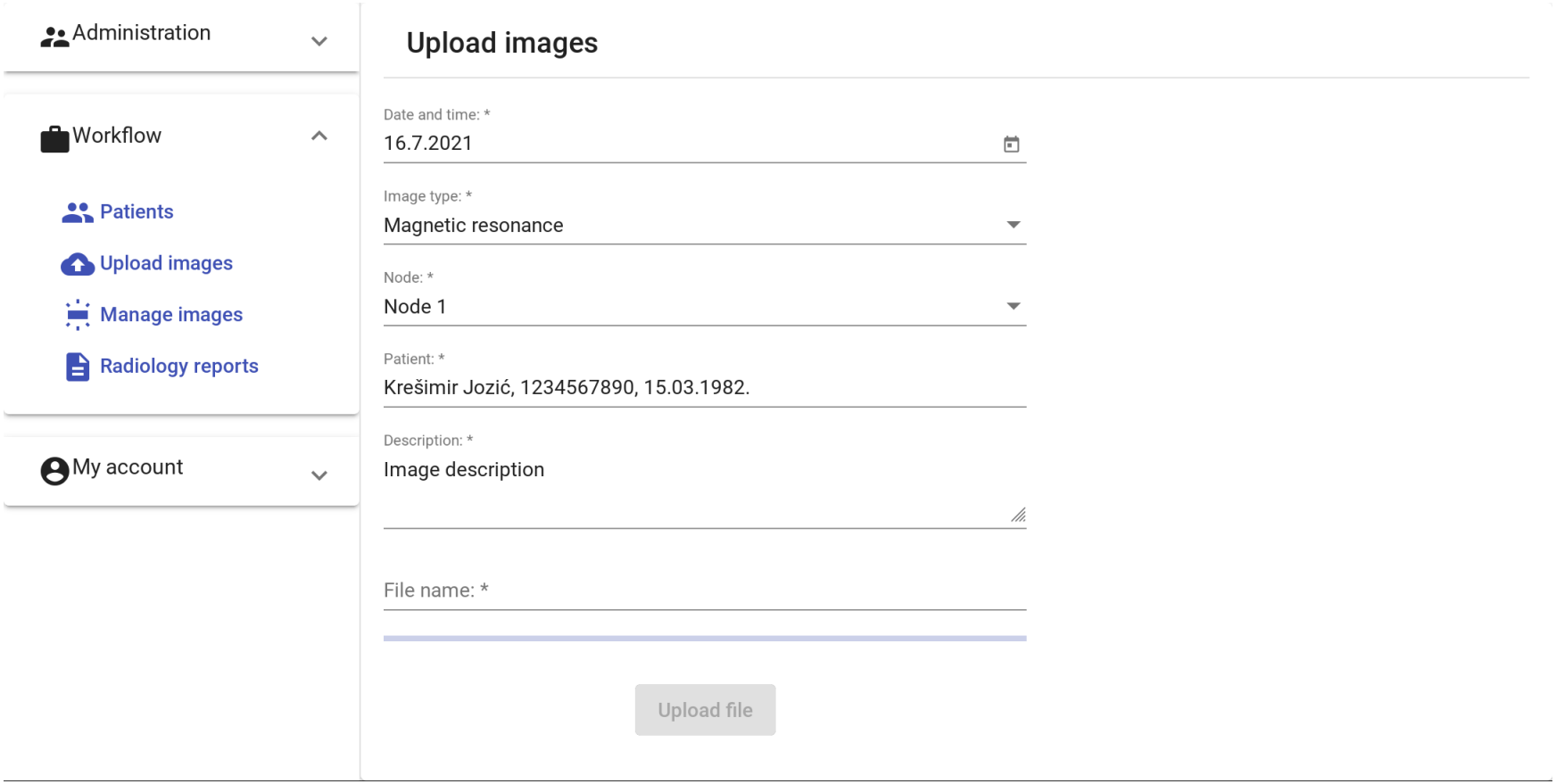
Slika C.4 – Upravljanje ustanovama



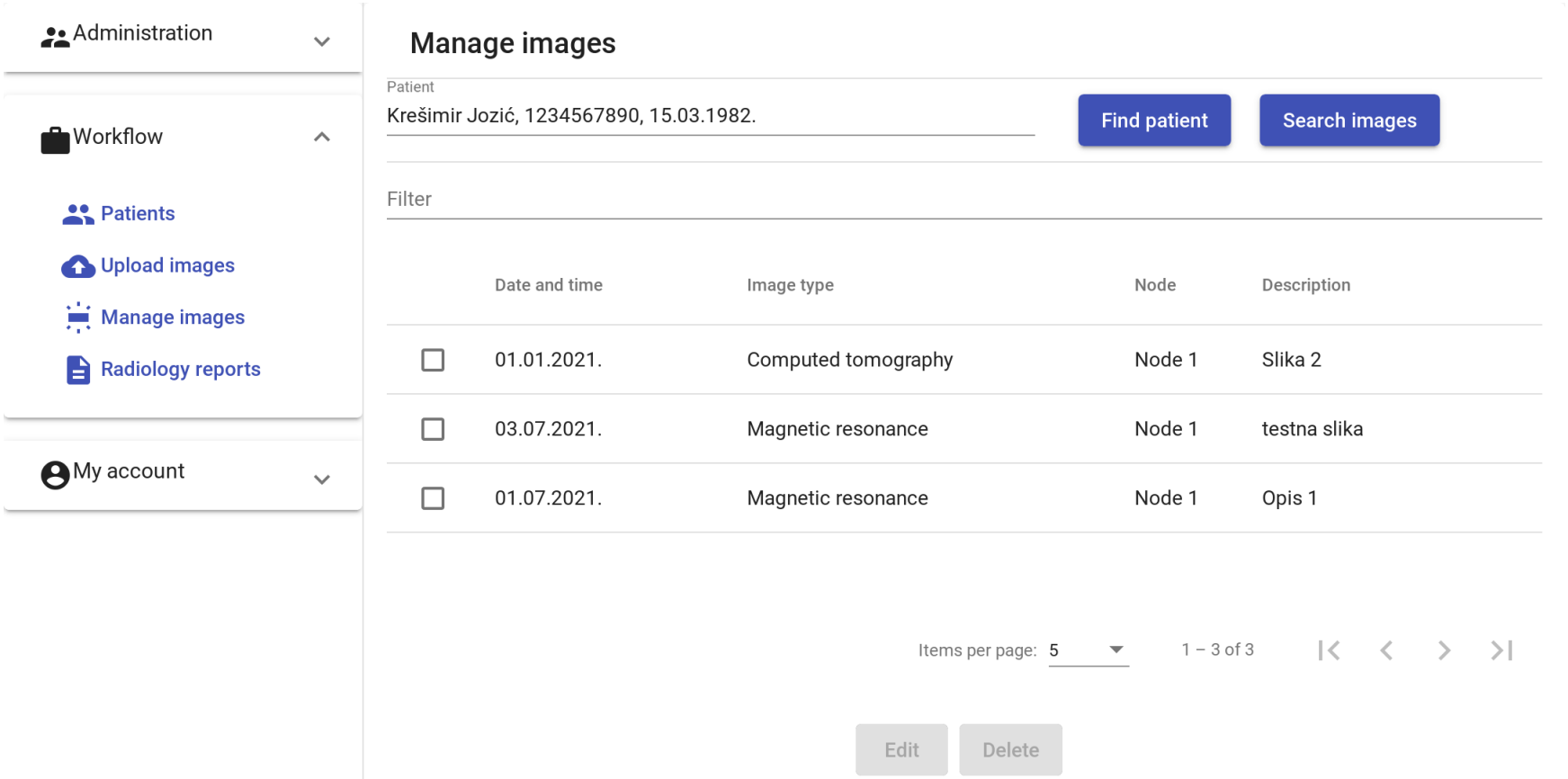
Slika C.5 – Uređivanje korisničkih podataka



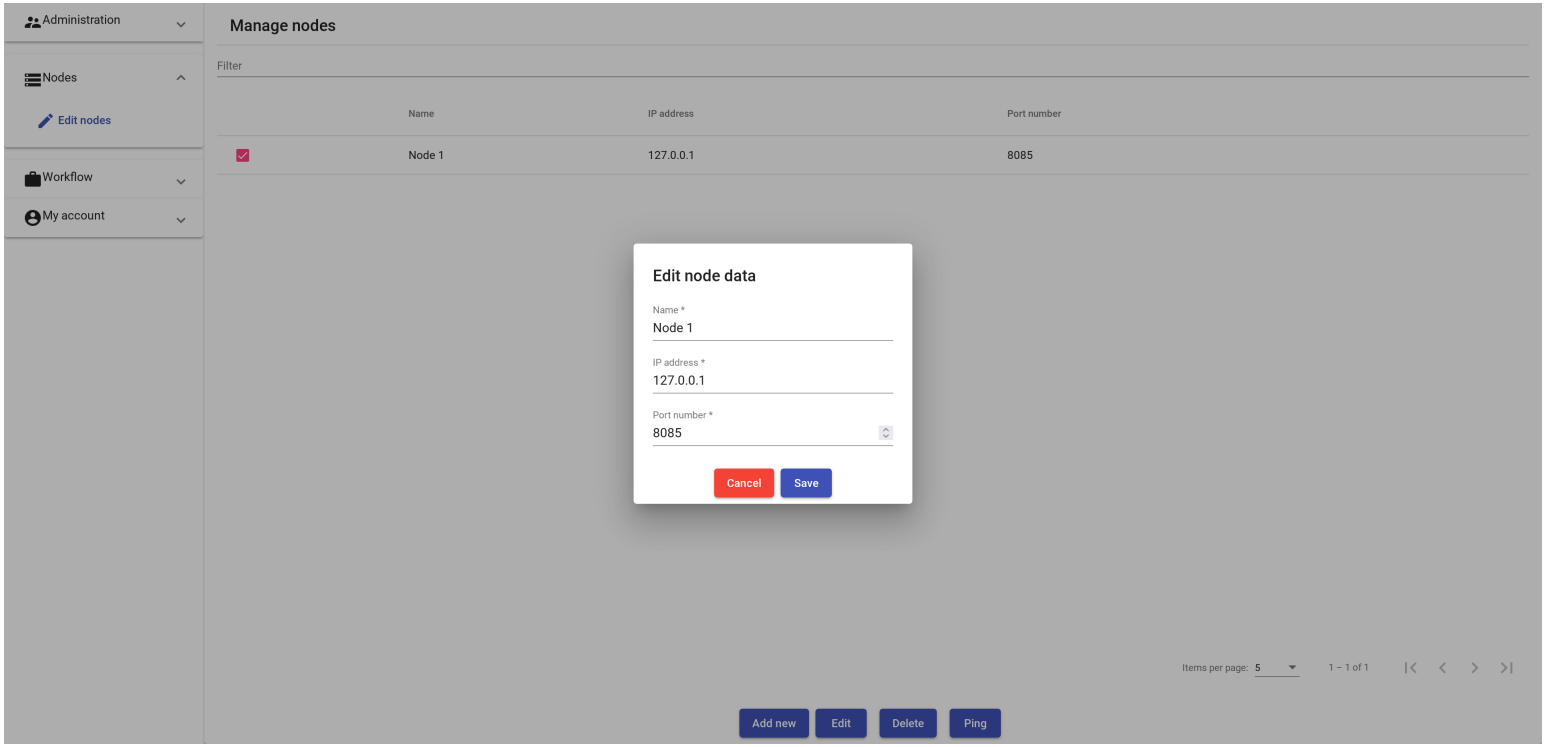
Slika C.6 – Upravljanje pacijentima



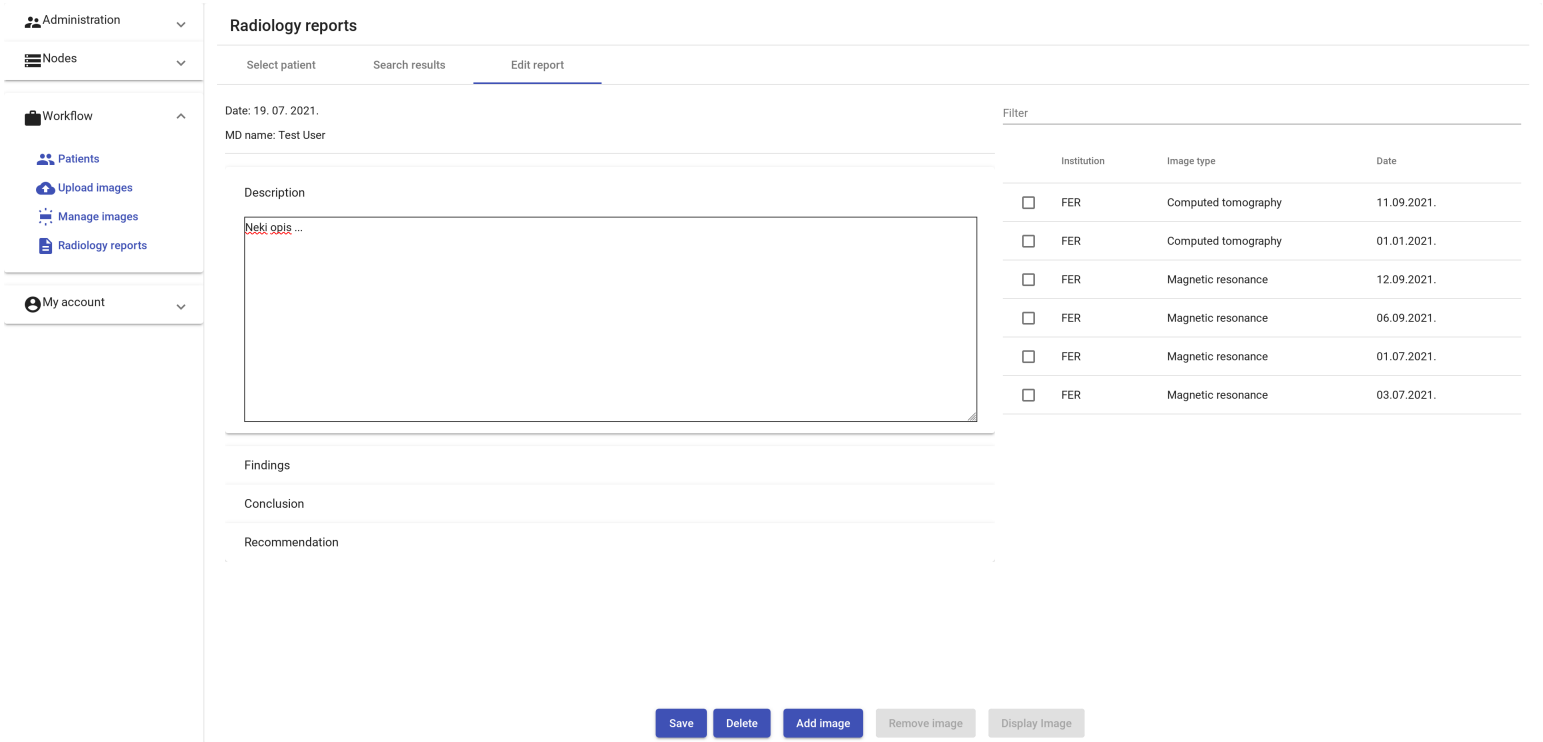
Slika C.7 – Slanje DICOM zapisa na poslužitelj



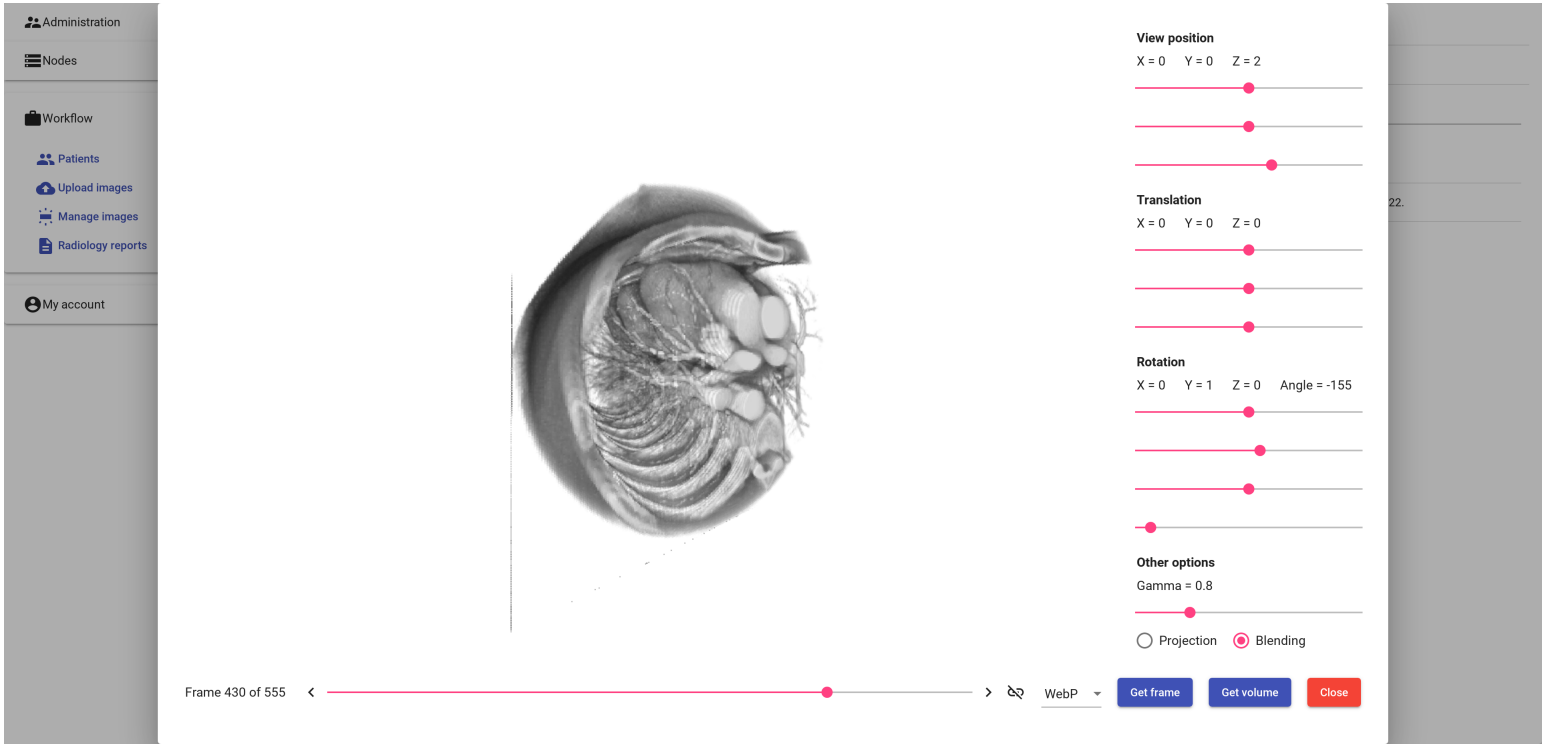
Slika C.8 – Upravljanje DICOM zapisima



Slika C.9 – Uređivanje čvora za iscrtavanje



Slika C.10 – Izrada radiološkog izvještaja



Slika C.11 – Pregledavanje DICOM zapisa

Literatura

[1] National Electrical Manufacturers Association (NEMA), American College of Radiology (ACR). Digital Imaging and Communications in Medicine, dostupno na: https://www.dicomstandard.org/ (pristupljeno: 18.09.2021.)

[2] Mustra, M., Delac, K., Grgic, M., "Overview of the DICOM standard", in 2008 50th International Symposium ELMAR, Vol. 1, 2008, str. 39–44.

[3] Koutelakis, G., Triantafyllou, G., Mandellos, G., Lymperopoulos, D., "A web PACS architecture based on WADO service of DICOM standard", 01 2005, str. 284–288.

[4] Graham, R., Perriss, R., Scarsbrook, A., "DICOM demystified: A review of digital file formats and their use in radiological practice", Clinical radiology, Vol. 60, 12 2005, str. 1133–40.

[5] Miller, R. B., "Response Time in Man-Computer Conversational Transactions", in Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I, ser. AFIPS '68 (Fall, part I). New York, NY, USA: Association for Computing Machinery, 1968, str. 267–277, dostupno na: https://doi.org/10.1145/1476589.1476628

[6] Vidal, F., Bello, F., Brodlie, K., John, N., Gould, D., Phillips, R., Avis, N., "Principles and Applications of Computer Graphics in Medicine", Computer Graphics Forum, Vol. 25, 03 2006, str. 113–137.

[7] Eklund, A., Dufort, P., Forsberg, D., LaConte, S., "Medical image processing on the GPU - Past, present and future", Medical image analysis, Vol. 17 8, 2013, str. 1073–94.

[8] Zhang, Q., Eagleson, R., Peters, T., "Volume Visualization: A Technical Overview with a Focus on Medical Applications", Journal of Digital Imaging, Vol. 24, 2010, str. 640–664.

[9] Zimmermann, K., Westermann, R., Ertl, T., Hansen, C., Weiler, M., "Level-of-Detail Volume Rendering via 3D Textures", in 2000 IEEE Symposium on Volume Visualization (VV 2000), 2000, str. 7–13.

[10] Marmitt, G., Friedrich, H., Slusallek, P., "Interactive Volume Rendering with Ray Tracing", in Eurographics 2006 - State of the Art Reports, Wyvill, B., Wilkie, A., (ur.). The Eurographics Association, 2006.

[11] Kainz, B., Grabner, M., Bornik, A., Hauswiesner, S., Muehl, J., Schmalstieg, D., "Ray Casting of Multiple Volumetric Datasets with Polyhedral Boundaries on Manycore GPUs", ACM Trans. Graph., Vol. 28, No. 5, Dec. 2009, str. 1–9, dostupno na: https://doi.org/10.1145/1618452.1618498

[12] Suwelack, S., Heitz, E., Unterhinninghofen, R., Dillmann, R., "Adaptive GPU Ray Casting Based on Spectral Analysis", in MIAR 2010 - 5th International Workshop on Medical Imaging and Augmented Reality, ser. Lecture Notes in Computer Science, Liao, H., Edwards, P. J. E., Pan, X., Fan, Y., Yang, G.-Z., (ur.), Vol. 6326. Beijing, China: Springer-Verlag, Sep. 2010, str. 169–178, oral session: Shape Modeling and Morphometry, dostupno na: https://hal.inria.fr/hal-00686463

[13] Gu, G., Kim, D., "Accurate and efficient GPU ray-casting algorithm for volume rendering of unstructured grid data", ETRI Journal, Vol. 42, 02 2020.

[14] Manke, F., Wünsche, B., "A Direct Volume Rendering Framework for the Interactive Exploration of Higher-Order and Multifield Data.", 01 2008, str. 199–206.

[15] Wheeler, G., Deng, S., Toussaint, N., Pushparajah, K., Schnabel, J., Simpson, J., Gómez, A., "Virtual interaction and visualisation of 3D medical imaging data with VTK and Unity", Healthcare Technology Letters, Vol. 5, 2018, str. 148–153.

[16] NVIDIA. RTX, dostupno na: https://www.nvidia.com/en-us/design-visualization/technologies/rtx/ (pristupljeno: 28.09.2021.)

[17] Noguera, J. M., Jiménez, J., Ogáyar, C., Segura, R., "Volume Rendering Strategies on Mobile Devices", in GRAPP/IVAPP, 2012.

[18] Schultz, C., Bailey, M., "Interacting with Large 3D Datasets on a Mobile Device", IEEE Computer Graphics and Applications, Vol. 36, No. 5, 2016, str. 19–23.

[19] Zhang, M., Schulze, J. P., "Server-Aided 3D DICOM Image Stack Viewer for Android Devices", Electronic Imaging, The Engineering Reality of Virtual Reality 2021, Jan. 2021, str. 179–1–179–7.

[20] Lee, K.-T., Lim, H.-J., Shin, S.-H., Kim, S.-W., Jang, M.-H., Ahn, Y.-H., Yoon, Y.-D., "An Internet-Based Telemadicine System", IJCSNS, Vol. 7, No. 1, 2007, str. 51, dostupno na: https://www.researchgate.net/profile/Sang_Wook_Kim2/publication/238085930_An_Internet-Based_Telemadicine_System/links/02e7e53c07ebbd78ce000000.pdf

[21] Thilagavath, K., Kavitha, M., “Sparkmed: A framework for Multimedia Medical Data Integration of Adaptive Mobile Object in Heterogeneous Systems”, International Journal of Science and Modern Engineering (IJISME), Vol. 1, No. 12, 2013, dostupno na: http://www.ijisme.org/attachments/File/v1i12/K04921011113.pdf

[22] Blazona, B., Mihajlovic, Z., “Visualization Service Based on Web Services”, in 2007 29th International Conference on Information Technology Interfaces, 2007, str. 673–678.

[23] Franco de Moraes, T., Amorim, P., Silva, J., Pedrini, H., “Web-based Interactive Visualization of Medical Images in a Distributed System”, 01 2019, str. 346–353.

[24] Min, Q., Wang, X., Huang, B., Xu, L., “Web-Based Technology for Remote Viewing of Radiological Images: App Validation”, Journal of medical Internet research, Vol. 22, 09 2020, str. e16224.

[25] Peter Eisert, Philipp Fechteler, “Remote Rendering of Computer Games”, in SIGMAP. INSTICC Press, 2007.

[26] Yoon, I., Neumann, U., “Web-Based Remote Rendering with IBRAC (Image-Based Rendering Acceleration and Compression)”, Computer Graphics Forum, Vol. 19, 10 2000.

[27] Lietsch, S., Marquardt, O., “A CUDA-Supported Approach to Remote Rendering”, in Proc. Int. Symp. on Visual Computing (ISVC), ser. Lecture Notes in Computer Science (LNCS), Vol. 4841. Springer, 2007, str. 724–733.

[28] Koller, D., Turitzin, M., Levoy, M., Tarini, M., Croccia, G., Cignoni, P., Scopigno, R., “Protected Interactive 3D Graphics via Remote Rendering”, ACM Trans. Graph., Vol. 23, No. 3, Aug. 2004, str. 695–703, dostupno na: https://doi.org/10.1145/1015706.1015782

[29] Belyaev, S., Chukanov, V., Shubnikov, V., “Bump Mapping for Isosurface Volume Rendering”, Journal of Image and Graphics, Vol. 5, 2017, str. 68–71.

[30] Strengert, M., Klein, T., Botchen, R. P., Stegmaier, S., Chen, M., Ertl, T., “Spectral volume rendering using GPU-based raycasting”, The Visual Computer, Vol. 22, 2006, str. 550–561.

[31] Patil, P., Patil, S., Chaudhari, D., “Conversion of DICOM Multi-Frame Medical Image into Multimedia Format using MATLAB”, International Journal of Engineering and Advanced Technology (IJEAT), Vol. 4, No. 2, Dec. 2014, str. 159–161.

[32] Stegmaier, S., Magallón, M., Ertl, T., “A Generic Solution for Hardware-Accelerated Remote Visualization”, in Proceedings of the Symposium on Data Visualisation 2002, ser. VISSYM ’02. Goslar, DEU: Eurographics Association, 2002, str. 87–ff.

[33] Paul, B., Ahern, S., Bethel, E. W., Brugger, E., Cook, R., Daniel, J., Lewis, K., Owen, J., Southard, D., "Chromium Renderserver: Scalable and Open Remote Rendering Infrastructure", IEEE transactions on visualization and computer graphics, Vol. 14, 06 2008, str. 627–39.

[34] Engel, K., Hastreiter, P., Tomandl, B., Eberhardt, K., Ertl, T., "Combining local and remote visualization techniques for interactive volume rendering in medical applications", in Proceedings Visualization 2000. VIS 2000 (Cat. No.00CH37145), 2000, str. 449–452.

[35] Tomandl, B. F., Hastreiter, P., Rezk-Salama, C., Engel, K., Ertl, T., Huk, W. J., Naraghi, R., Ganslandt, O., Nimsky, C., Eberhardt, K. E. W., "Local and Remote Visualization Techniques for Interactive Direct Volume Rendering in Neuroradiology", RadioGraphics, Vol. 21, No. 6, Nov. 2001, str. 1561–1572, dostupno na: http://pubs.rsna.org/doi/10.1148/radiographics.21.6.g01nv241561

[36] Hassan, A. H., Fluke, C. J., Barnes, D. G., "A Distributed GPU-Based Framework for Real-Time 3D Volume Rendering of Large Astronomical Data Cubes", Publications of the Astronomical Society of Australia, Vol. 29, No. 3, 2012, str. 340–351, dostupno na: http://dx.doi.org/10.1071/AS12025

[37] Nyström, I., Malmberg, F., Vidholm, E., Bengtsson, E., "Segmentation and Visualization of 3D Medical Images through Haptic Rendering", in Proceedings of the 10th International Conference on Pattern Recognition and Information Processing (PRIP 2009). Publishing Center of BSU, 2009, str. 43–48, dostupno na: http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-107657

[38] Arbelaiz., A., Moreno., A., Kabongo., L., V. Diez., H., García Alonso., A., "Interactive Visualization of DICOM Volumetric Datasets in the Web - Providing VR Experiences within the Web Browser", in Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - IVAPP, (VISIGRAPP 2017), INSTICC. SciTePress, 2017, str. 108–115.

[39] Karonis, N., Papka, M., Binns, J., Bresnahan, J., Insley, J., Jones, D., Link, J., "High-resolution remote rendering of large datasets in a collaborative environment", Future Generation Computer Systems, Vol. 19, 08 2003, str. 909–917.

[40] Sondur, S., Kant, K., Vucetic, S., Byers, B., "Storage on the Edge: Evaluating Cloud Backed Edge Storage in Cyberphysical Systems", in 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2019, str. 362–370.

[41] OpenAPI Initiative. OpenAPI specification, dostupno na: https://swagger.io/specification/ (pristupljeno: 23.09.2021.)

[42] Clark Evans, Ingy döt Net, Oren Ben-Kiki. YAML Ain't Markup Language, dostupno na: https://yaml.org/ (pristupljeno: 26.09.2021.)

[43] Internet Engineering Task Force (IETF). Hypertext Transfer Protocol Version 2 (HTTP/2), dostupno na: https://datatracker.ietf.org/doc/html/rfc7540 (pristupljeno: 18.09.2021.)

[44] Internet Engineering Task Force (IETF). Hypertext Transfer Protocol Version 3 (HTTP/3), dostupno na: https://datatracker.ietf.org/doc/draft-ietf-quic-http/ (pristupljeno: 18.09.2021.)

[45] Internet Engineering Task Force (IETF). QUIC, dostupno na: https://quicwg.org/ (pristupljeno: 23.09.2021.)

[46] Internet Engineering Task Force (IETF). JSON Web Token (JWT), dostupno na: https://datatracker.ietf.org/doc/html/rfc7519 (pristupljeno: 18.09.2021.)

[47] Ćurguz, J., "Vulnerabilities of the SSL/TLS Protocol", Vol. 6, 05 2016, str. 245–256.

[48] OpenZFS Project. OpenZFS, dostupno na: https://openzfs.org/wiki/Main_Page (pristupljeno: 18.09.2021.)

[49] Khronos group. OpenGL, dostupno na: https://www.opengl.org/ (pristupljeno: 24.09.2021.)

[50] Zhou, Z.-H., Wen, X.-J., "3D Reconstruction of medical image based on opengl", Journal of Computers (Taiwan), Vol. 29, 04 2018, str. 249–260.

[51] Kruger, J., Westermann, R., "Acceleration techniques for GPU-based volume rendering", in IEEE Visualization, 2003. VIS 2003., 2003, str. 287–292.

[52] Raspe, M., Müller, S., "Controlling GPU-based Volume Rendering using Ray Textures", in WSCG'2008, 2008, str. 277–283.

[53] Philip Rideout. Single-Pass Raycasting, dostupno na: https://prideout.net/blog/old/blog/index.html@p=64.html (pristupljeno: 25.09.2021.)

[54] Martino Pilia. GPU-accelerated single-pass volumetric raycasting in Qt and OpenGL, dostupno na: https://martinopilia.com/posts/2018/09/17/volume-raycasting.html (pristupljeno: 25.09.2021.)

[55]Google. Angular, dostupno na: https://angular.io/ (pristupljeno: 18.09.2021.)

[56] Google. Angular Material, dostupno na: https://material.angular.io/ (pristupljeno: 26.09.2021.)

[57] Microsoft. TypeScript, dostupno na: https://www.typescriptlang.org/ (pristupljeno: 26.09.2021.)

[58] Google. WebP, dostupno na: https://developers.google.com/speed/webp (pristupljeno: 18.09.2021.)

[59]Google. Go, dostupno na: https://golang.org/ (pristupljeno: 18.09.2021.)

[60] Gin-Gonic. Gin, dostupno na: https://github.com/gin-gonic/gin (pristupljeno: 26.09.2021.)

[61]Jinzhu. GORM, dostupno na: https://gorm.io/index.html (pristupljeno: 26.09.2021.)

[62] Eric Woroshow et al. OpenGL with Golang, dostupno na: https://github.com/go-gl (pristupljeno: 26.09.2021.)

[63] Suyash Kumar. High Performance DICOM Medical Image Parser in Go, dostupno na: https://github.com/suyashkumar/dicom (pristupljeno: 26.09.2021.)

[64] Kolesa Group. Simple and fast webp library for golang, dostupno na: https://github.com/kolesa-team/go-webp (pristupljeno: 26.09.2021.)

[65] Lucas Clemente. A QUIC implementation in pure go, dostupno na: https://github.com/lucas-clemente/quic-go (pristupljeno: 26.09.2021.)

[66] Daniel Farina. Pure Go Postgres driver for database/sql, dostupno na: https://github.com/lib/pq (pristupljeno: 26.09.2021.)

[67] go-yaml. YAML support for the Go language, dostupno na: https://github.com/go-yaml/yaml/tree/v3 (pristupljeno: 26.09.2021.)

[68] Luis Gabriel Gomez, Michael Fridman, Alistair Hey. Golang implementation of JSON Web Tokens (JWT), dostupno na: https://github.com/golang-jwt/jwt (pristupljeno: 26.09.2021.)

[69] Bo-Yi Wu. JWT Middleware for Gin framework, dostupno na: https://github.com/appleboy/gin-jwt (pristupljeno: 26.09.2021.)

[70] Cockroach Labs. CockroachDB, dostupno na: https://www.cockroachlabs.com/ (pristupljeno: 18.09.2021.)

[71] David Clunie. David Clunie's Medical Image Format Site, dostupno na: http://www.dclunie.com/ (pristupljeno: 29.09.2021.)

[72] OFFIS DICOM Toolkit DCMTK. DICOM Test Images, dostupno na: https://support.dcmtk.org/redmine/projects/dcmtk/wiki/DICOM_images (pristupljeno: 29.09.2021.)

[73] GDCM : Grassroots DICOM library. Sample DataSet, dostupno na: http://gdcm.sourceforge.net/wiki/index.php/Sample_DataSet (pristupljeno: 29.09.2021.)

[74] Moving Picture Experts Group . High Efficiency Image File Format, dostupno na: https://mpeg.chiariglione.org/standards/mpeg-h/image-file-format (pristupljeno: 29.09.2021.)

[75] Alliance for Open Media. AV1 Image File Format, dostupno na: https://aomediacodec.github.io/av1-avif/ (pristupljeno: 29.09.2021.)

[76]ISO/IEC. JPEG XL, dostupno na: https://jpeg.org/jpegxl/ (pristupljeno: 29.09.2021.)

[77] Khronos group. WebGL, dostupno na: https://www.khronos.org/webgl/ (pristupljeno: 26.09.2021.)

[78] Khronos group. OpenGL ES, dostupno na: https://www.khronos.org/opengles/ (pristupljeno: 26.09.2021.)

[79] Khronos group. Vulkan, dostupno na: https://www.vulkan.org/ (pristupljeno: 26.09.2021.)

[80] Khronos group. Open Computing Language, dostupno na: https://www.khronos.org/opencl/ (pristupljeno: 26.09.2021.)

Popis simbola

Skraćenice

ACRAmerican College of Radiology

AVIFAV1 Image File Format

AZDOApproaching Zero Driver Overhead

CgC for graphics

CLICommand Line Interface

CTComputed Tomography

CUDACompute Unified Device Architecture

DICOMDigital Imaging and Communications in Medicine

FFTFast Fourier transform

HEIFHigh Efficiency Image File Format

HTTPHypertext Transfer Protocol

HTTPSHypertext Transfer Protocol Secure

HUHounsfield Unit

JSONJavaScript Object Notation

JWTJSON Web Tokens

LANLocal Area Network

MIPMultum In Parvo

MRMagnetic Resonance

NASNetwork Attached Storage

NEMANational Electrical Manufacturers Association

NoSQLNon-SQL

OpenCLOpen Computing Language

OpenGLOpen Graphics Library

OpenGL ESOpenGL for Embedded Systems

PACSPicture Archiving and Communication Systems

PETPositron Emission Tomography

RAIDRedundant Array of Inexpensive Disks

RESTRepresentational state transfer

SANStorage Area Network

SQLStructured Query Language

SSLSecure Sockets Layer

TCPTransmission Control Protocol

TLSTransport Layer Security

UDPUser Datagram Protocol

USUltrasound

WebGLWeb Graphics Library

YAMLYet Another Markup Language

ZFSZettabyte File System

Popis slika

Popis tablica

Životopis

Krešimir Jozić rođen je 15. ožujka 1982. godine u Vukovaru, Hrvatska. Godine 2000. upisuje prediplomski studij računarstva na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Studij završava 2006. godine diplomskim radom: „Programirljivo grafičko sklopovlje".

Od siječnja do srpnja 2007. g radi kako sistemski inženjer na IBM mainframe računalima u Zagrebačkoj banci. U periodu od srpnja 2007. g. do lipnja 2019. g. radi u INI na više radnih mjesta u području industrijskih mjerenja i automatizacije. Od lipnja 2019. g. do prosinca 2019. radi u Siemensu kao inženjer za instrumentaciju. Odvajanjem energetskog odjela Siemensa prelazi u tvrtku Siemens Energy gdje radi do danas.

Njegovi istraživački interesi obuhvaćaju područje računalne grafike s primjenom u medicini. Do sada je objavio 1 poglavlje u knjizi, 7 radova na međunarodnim znanstvenim skupovima i 2 rada u časopisima.

Popis objavljenih djela

Poglavlja u knjigama

1. A. Jovic, K. Jozic, D. Kukolja, K. Friganovic, M. Cifrek, „Challenges in Designing Software Architectures for Web Based Biomedical Signal Analysis", In: "Medical Big Data and Internet of Medical Things Advances, Challenges and Applications". A. E. Hassanien, N. Dey, S. Borra (Eds.), CRC Press Taylor & Francis Group, Boca Raton, pp. 81-112, 2018.

Radovi u časopisima

1. K. Jozić, N. Frid, A. Jović, Ž. Mihajlović, „DICOM SIVR: A web architecture and platform for seamless DICOM image and volume rendering", SoftwareX, Volume 18, 2022.
2. K. Jozić, Ž. Mihajlović, „Programirljivo grafičko sklopovlje", Automatika: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije, 2006, 47 (3-4), 177-181.

Radovi na međunarodnim znanstvenim skupovima

1. K. Jozic, A. Jovic, Z. Mihajlovic, „Seamless Remote Rendering of DICOM Images ”, Proceedings of ICACTE 2021, International Conference on Advanced Computer Theory and Engineering, 2021.
2. A. Jovic, D. Kukolja, K. Friganovic, K. Jozic, M. Cifrek, „MULTISAB: A Web Platform for Analysis of Multivariate Heterogeneous Biomedical Time-Series”, Proceedings of IUPESM 2018, IFMBE Proceedings Volume 68/1, World Congress on Medical Physics and Biomedical Engineering 2018, pp. 411-415, 2018.
3. A. Jovic, K. Jozic, D. Kukolja, K. Friganovic, M. Cifrek, „Parallelization in biomedical time series analysis web platform: the MULTISAB project experience”, Proceedings of 41th International Convention MIPRO 2018, pp. 337-342, 2018.
4. A. Jovic, D. Kukolja, K. Friganovic, K. Jozic, S. Car, „Biomedical time series preprocessing and expert system based feature extraction in MULTISAB platform”, Proceedings of 40th Jubilee International Convention MIPRO 2017, pp. 349-354, 2017.
5. K. Friganovic, A. Jovic, K. Jozic, D. Kukolja, M. Cifrek, „MULTISAB project: a web platform based on specialized frameworks for heterogeneous biomedical time series analysis - an architectural overview”, Proceedings of the International Conference on Medical and Biological Engineering, IFMBE vol. 62, CMBEBIH 2017, pp. 9-15, 2017.
6. A. Jovic, D. Kukolja, K. Jozic, M. Cifrek, „Use Case Diagram Based Scenarios Design for a Biomedical Time-Series Analysis Web Platform”, Proceedings of the 39th International Convention MIPRO 2016, pp. 326-331, 2016.
7. A. Jovic, D. Kukolja, K. Jozic, M. Horvat, „A Web Platform for Analysis of Multivariate Heterogeneous Biomedical Time-Series - a Preliminary Report”, Proceedings of the 23rd International Conference on Systems, Signals and Image Processing (IWSSIP 2016), p. 70, 2016.

Biography

Krešimir Jozić was born on March 15, 1982 in Vukovar, Croatia. In 2000, he enrolled in the undergraduate study of computer science at the Faculty of Electrical Engineering and Computing, University of Zagreb. He completed his studies in 2006 with a diploma thesis: "Programmable Graphics Hardware".

From January to July 2007, he worked as a systems engineer on IBM mainframe computers at Zagrebačka banka. In the period from July 2007 to June 2019, he worked at INA in several positions in the field of industrial measurements and automation. From June 2019 to December 2019, he worked at Siemens as an instrumentation engineer. By separating the energy department of Siemens, he transferred to the company Siemens Energy, where he works to this day.

His research interests include the field of computer graphics with application in medicine. So far, 1 chapter in literature, 7 papers at international scientific conferences, and 2 papers in a journal have been published.