Exam Hall Seat Allocation System

PRESENTED BY
Alimon N A - MUT20CS024
Arjun P Unni - MUT20CS047
Gowri M - MUT20CS070
Sudhi Krishna N A - MUT20CS116

GUIDED BY
Steffy Livera

Contents

Problem Statement

The goal is to create a web-based system to manage exam seating arrangements for MITS, which will digitize the exam seat allocation on campus.

Introduction

- Exam Hall Seat Allocation System - the perfect solution to automate seat allocation, and to save time and effort for exam cell officials.
- With the power of technology, the system streamlines the process and makes the job easier for everyone involved.
- Say goodbye to manual allocation and hello to efficiency.

Current System

The existing practice is the arrangement of seats for examinations manually by the exam cell faculty, with registered students and exam details provided by the university.

DRAWBACKS

- Time Consuming
- Requires a lot of effort and resources
- Not scalable and lacks flexibility
- Does not provide any real-time information

Literature Review

Sl No.	Title	Author	Methodology
1	Algorithm for efficient seat allocation process in college exam system	Prof.Nilima Nikam, Akshata Jagdale, Gunjan Patil, Prachi Patil	• Data used: Room information, Exam information. • Calculated the number of columns and rows for each column. • Assigned columns to subjects • Assigned rooms and column numbers to the subjects based on the column size.
2	An Efficient Heuristic for Exam Seat Allocation	S. S. Ali, H. S. Al-Rawi	• The heuristic comprises of two phases, • Greedy strategy to assign seats to students based on their preferences and constraints • Local search strategy to improve the initial allocation by considering different types of moves. • This strategy iteratively applies these moves to the initial allocation and selects the move that results in the best improvement in the objective function.

3	Developing Examination Management System: Senior Capstone Project	S. Vasupongayya, W. Noodam, and P. Kongyong	• The algorithm used here is the greedy algorithm with a simple priority function. • The algorithm sorts subjects based on the number of students in the subject • Assigned rooms for morning and afternoon sections of each day • Mapped large subjects to large rooms while also taking into account the room capacity • If a subject cannot fit in one room, the algorithm searches for another room to assign the remaining seats. • The algorithm continues until all rooms are filled.
4	Algorithm For Efficient Seating Plan For Centralized Exam System	Prosanta Kumar Chaki, Shikha Anirban	• The program needs to know about the rooms and exams. • The program tries to prevent cheating by keeping a distance between students. • The program calculates the number of seats needed and the number of extra seats. • The program finds empty rooms to accommodate extra students. • The program assigns columns to each subject, calculating the number of columns and the maximum usable column. • The program assigns seats to each subject, maintaining a distance between the same subjects.

Proposed System

An automated exam seat arrangement system that aims to streamline the process of assigning seats for students taking exams.

METHODOLOGY

- Gather requirements from Exam cell
- Design a user-friendly interface
- Develop an algorithm for seat allocation
- Ensure data security and privacy

RELEVANCE

- Increase the efficiency and accuracy of exam seat arrangement process
- Providing a better experience for both students and exam administrators.

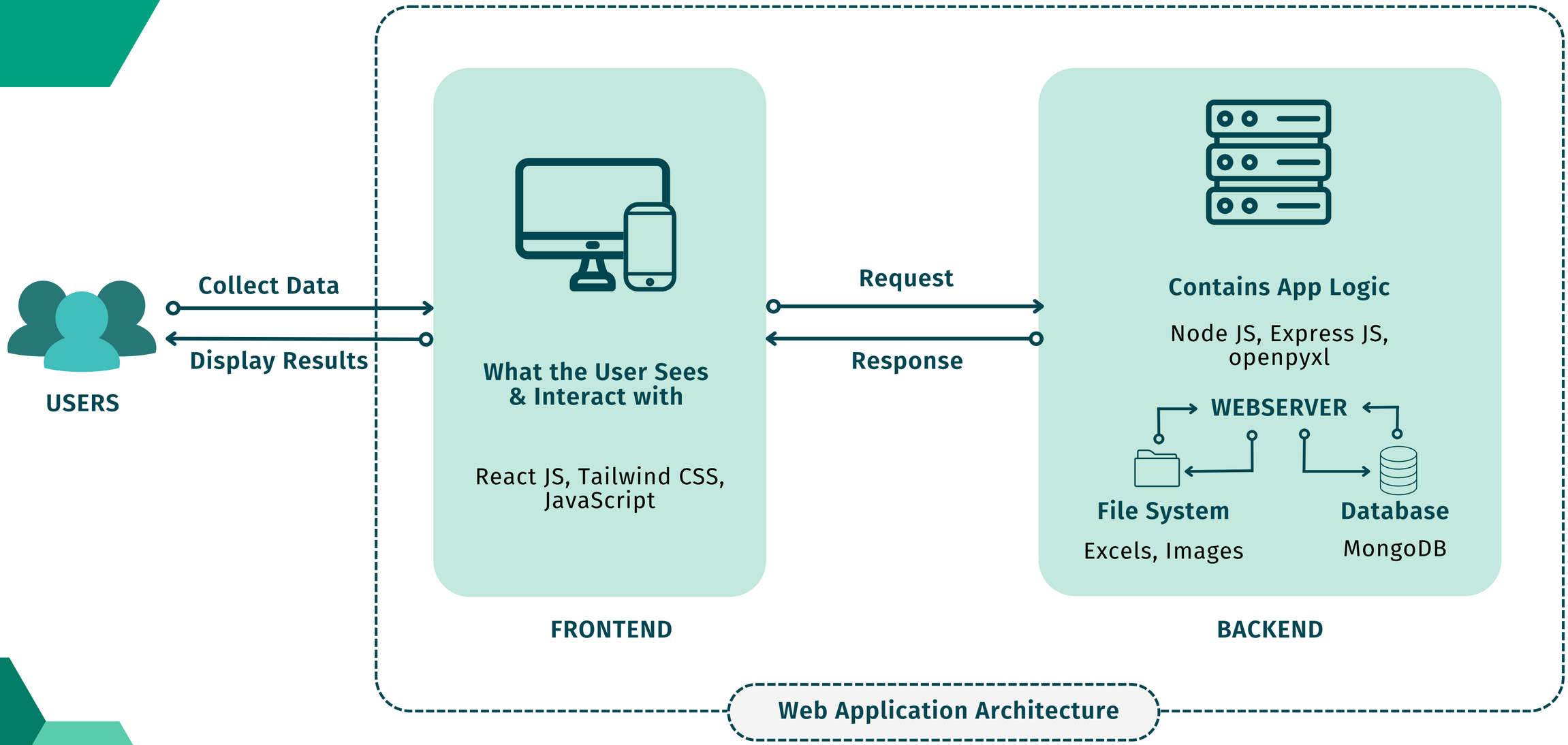
Advantages

Efficiently allocating seats to students for exams.

Alleviate the administrative workload for exam administrators.

Created a scalable solution that efficiently accommodates large numbers of students and multiple exam halls.

Architecture Diagram



Module Details

Login/Register : Users can login and can even create new accounts.

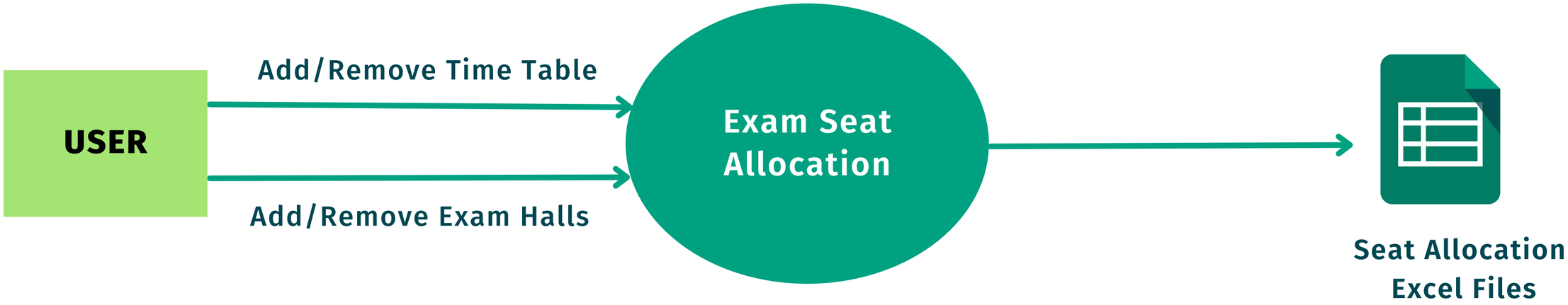
Manage Rooms : Add or remove exam halls and the number of available seats in each hall

Exam Details : Update or view upcoming exam details of each semester

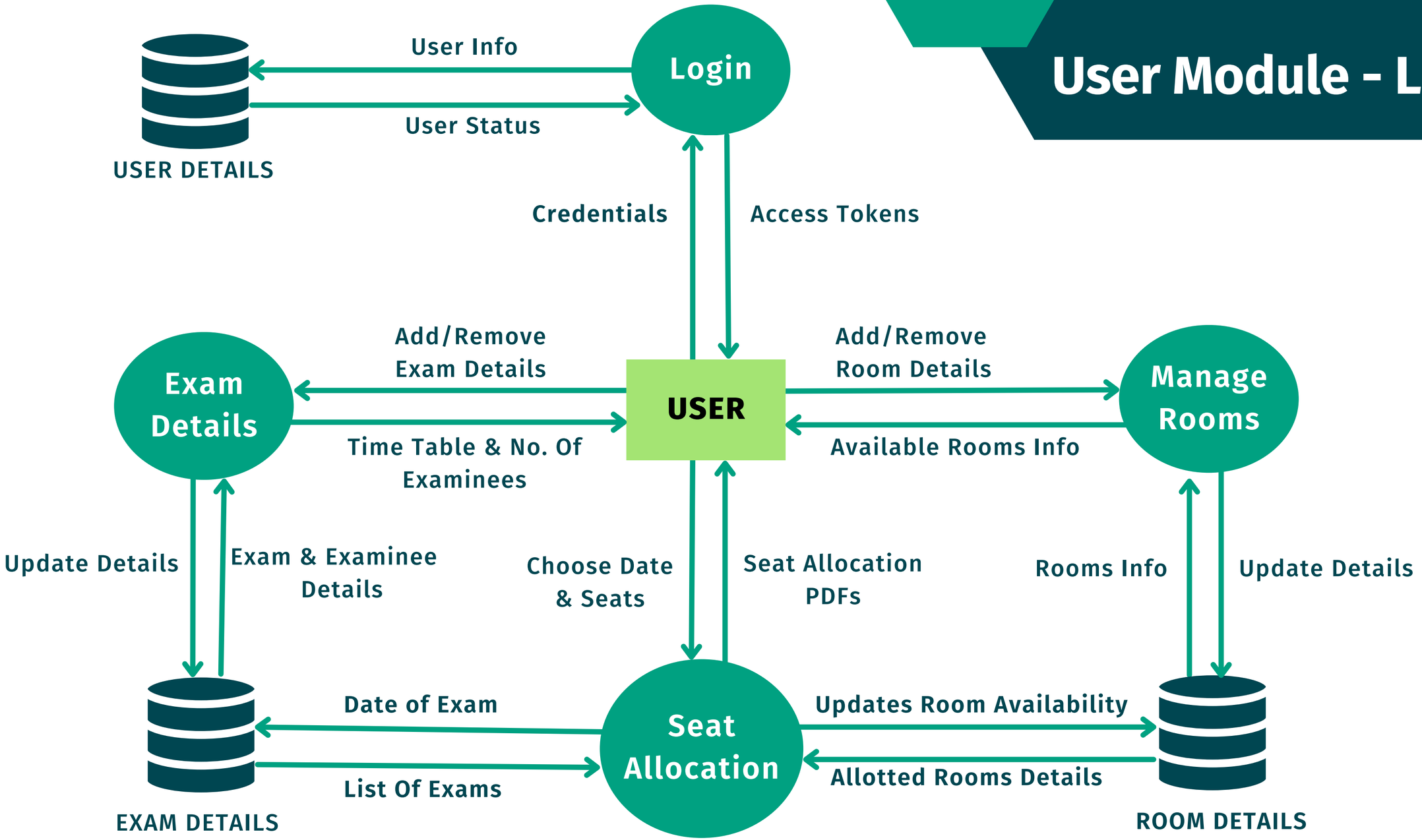
Seat Allocation : Generate excel sheets with exam halls and participants details

System Design - DFD

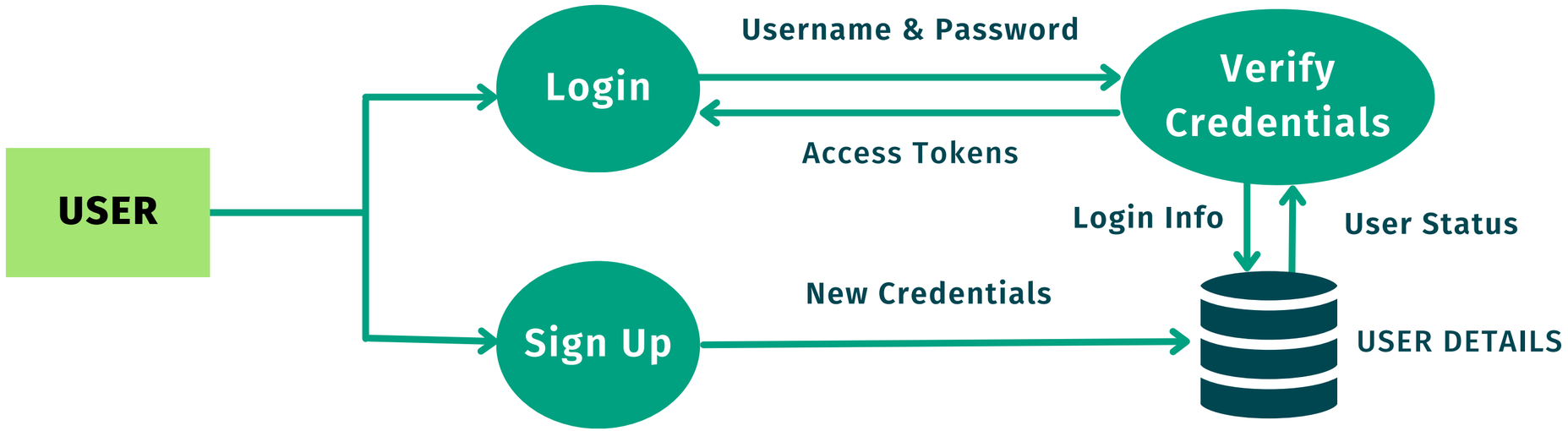
User Module - Level 0



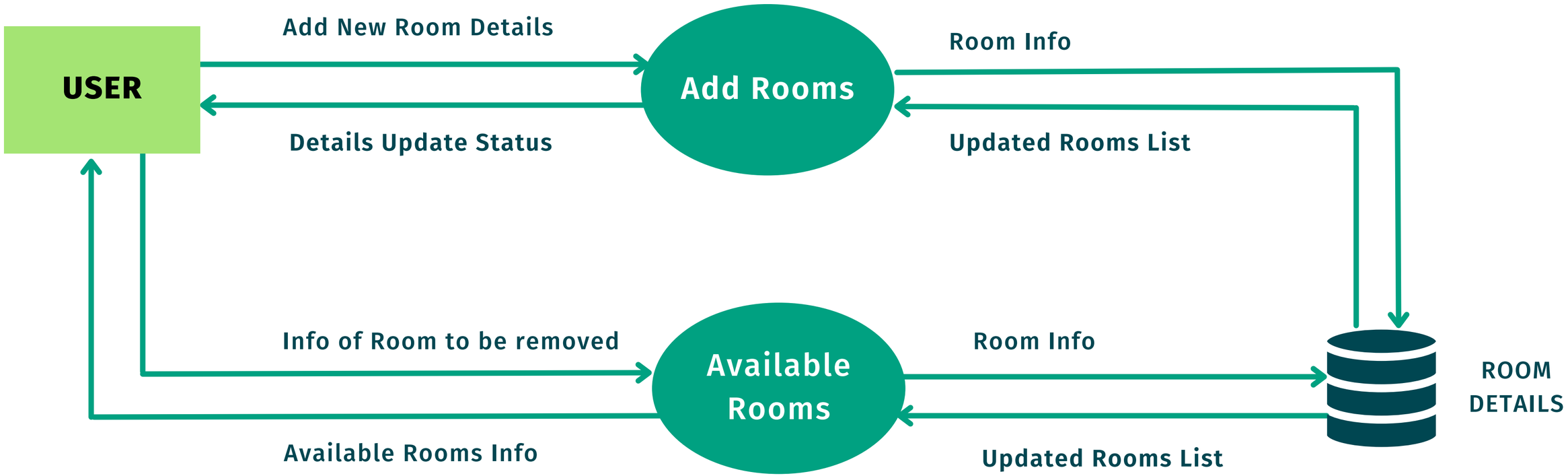
User Module - Level 1



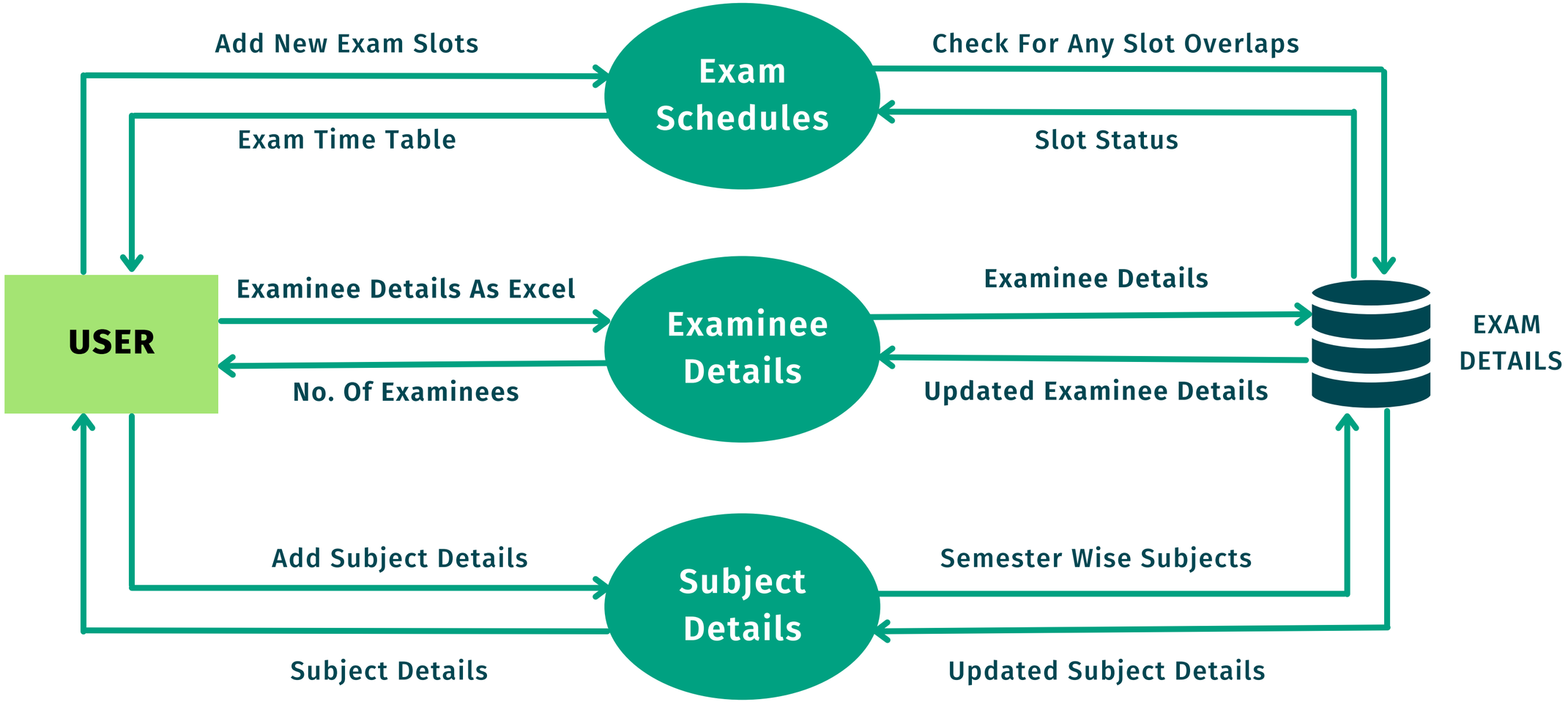
Login


USER
Login
Sign Up
Username & Password
Access Tokens
Verify Credentials
Login Info
User Status
New Credentials
USER DETAILS

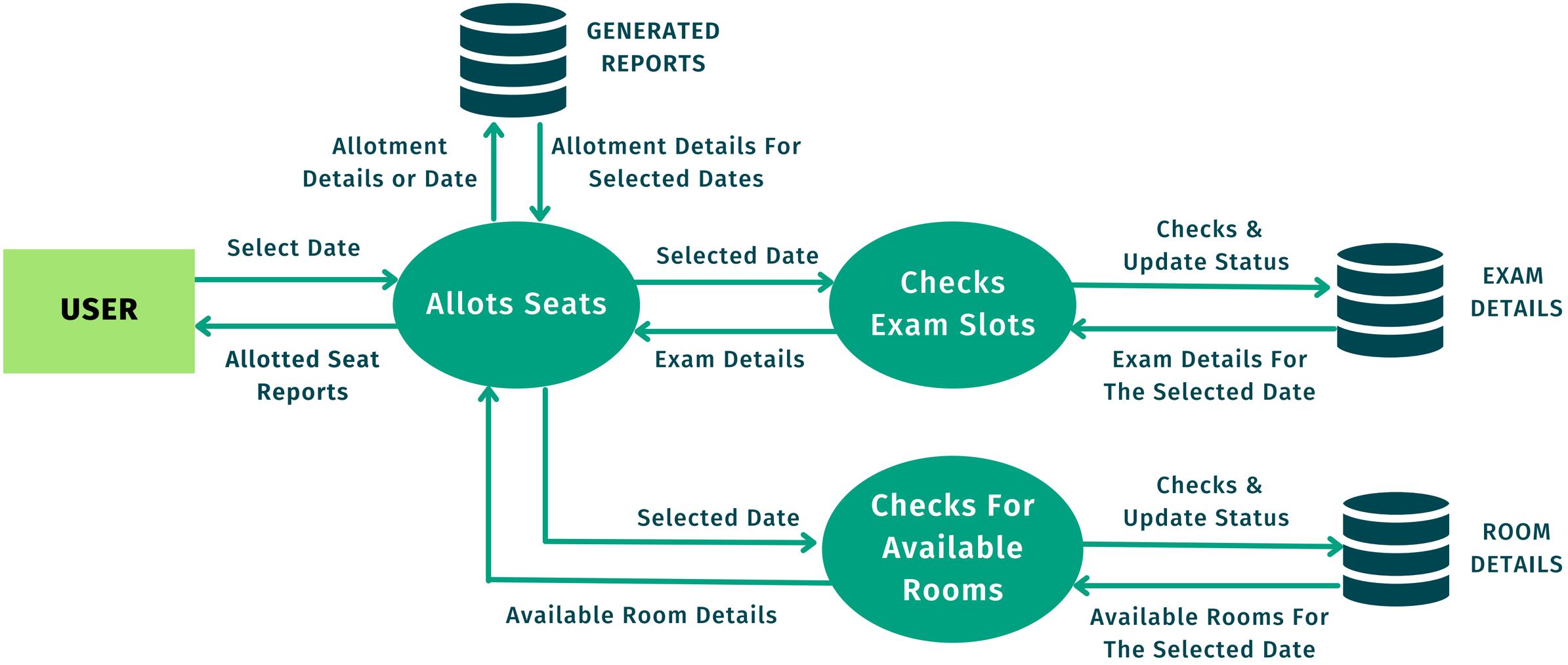
Manage Rooms



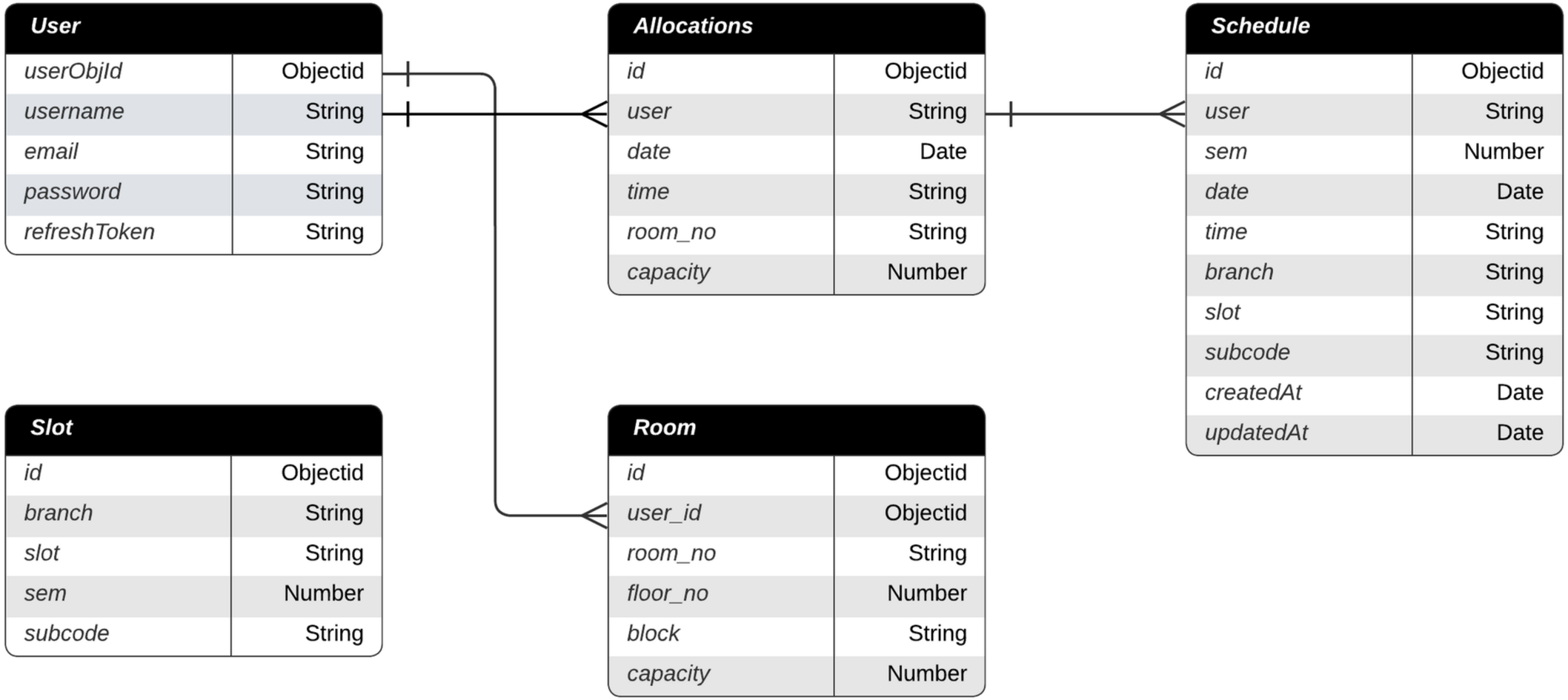
Exam Details



Seat Allocation



Database Table Design



System Requirements

To run the Exam Seat Allocation web application that takes Excel sheets as input on a local server the following tools are required.

SOFTWARE REQUIREMENTS

- Web Browser
- Node JS
- Python

HARDWARE REQUIREMENTS

- High Performance computer
- Sufficient Network Bandwidth

Technologies Used

MongoDB

Express JS

React JS

Node JS

openpyxl

Implementation Details

React JS

- Developed the user interface using HTML, CSS, and React.js.
- Utilized Axios library for making HTTP requests to the server.
- Design responsive web pages for optimal user experience.
- Implement protected routing using React Router v6.

App.jsx

C: > Users > alimo > OneDrive > Documents > GitHub > Exam-Seat-Arrangement-System > src > App.jsx

```jsx
import Login from "./components/Login";
import Register from "./components/Register";
import Layout from "./components/Layout";
import Home from "./components/Home";
import ManageRoom from "./components/ManageRoom";
import UniversityExam from "./components/UniversityExam";
import SeatAllocation from "./components/SeatAllocation";
import { Route, Routes } from 'react-router-dom';
import RequireAuth from "./components/RequireAuth";
import PersistLogin from "./components/PersistLogin";

function App() {
  return (
    <Routes>
      {/* public routes */}
      <Route path="/" element={<Login />} />
      <Route path="/register" element={<Register />} />

      {/* Protection needed routes */}
      <Route element={<PersistLogin />}>
        <Route element={<RequireAuth />}>
          <Route path="/" element={<Layout />}>
            <Route path="home" element={<Home />} />
            <Route path="manage-room" element={<ManageRoom />} />
            <Route path="university-exam" element={<UniversityExam />} />
            <Route path="seat-allocation" element={<SeatAllocation />} />
          </Route>
        </Route>
      </Route>
    </Routes>
  );
}

export default App;
```

Tailwind CSS

- Rapidly build UI by applying utility classes directly to HTML elements.
- Highly customizable through a configuration file for defining colors, spacing, breakpoints, etc.
- Include CSS file or import it, and integrate with build tools for optimization and customization.

tailwind.config.js

C: > Users > alimo > OneDrive > Desktop > tailwind.config.js > ...

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [
    "./src/**/*.{js,jsx,ts,tsx}",
  ],
  theme: {
    extend: {
      screens: {
        st: '1384px',
        'hw': { 'max': '1384px' }
      },
      colors: {
        'green-login': '#92DBB6',
        'green-medium': '#003944',
        'green-light': '#295D68',
        'green-dark': '#111F35',
        'green-save': '#23CA85',
        'grey-all': '#BECCCF',
        'red-download': '#DD7A96',
        'background': '#F2F4E7',
      },
      letterSpacing: {
        needed: '.09em',
      },
      fontFamily: { //custom font style
        'Outfit-Thin': "Outfit-Thin",
        'Outfit-ExtraLight': "Outfit-ExtraLight",
        'Outfit-Light': "Outfit-Light",
        'Outfit-Regular': "Outfit-Regular",
```

```
<div className="flex flex-col h-full py-5">
    <NavLink to="home" className={({ isActive }) => isActive ? "bg-green-dark py-2 w-full flex flex-row" : "hover:bg-green-light py-2 w-full flex flex-row"}>
        <p className={`font-Outfit-Medium tracking-needed text-white ${expand ? "mx-6" : "absolute left-[-999px]"}`}>Home</p>
        <img src={home} alt="H" className={`${expand ? "absolute left-[-999px]" : "h-7 w-7 ml-3 mr-4 my-1"}`} title="home" />
    </NavLink>
```

Node JS

- Use Node.js as the server-side language for the backend implementation.
- Utilize Express.js as the web application framework for handling routes and middleware.
- Incorporate frameworks like Cors, Mongoose, Nodemailer, Express FileUpload, and CookieParser to enhance functionality and streamline development.

server.js

C: > Users > alimo > OneDrive > Desktop > server.js > ...

```
const connectDB = require('./config/connectDB');
const cookieParser = require('cookie-parser');
const cors = require('cors');
const corsOptions = require('./config/corsOptions');
const credentials = require('./middlewares/credentials');
const express = require('express');
const mongoose = require('mongoose');
const verifyJWT = require('./middlewares/verifyJWT');
require('dotenv').config();
const app = express();
const PORT = process.env.PORT || 5500;
connectDB();
// Handle options credentials check - before CORS!
// and fetch cookies credentials requirement
app.use(credentials);
// Cross Origin Resource Sharing
app.use(cors(corsOptions));

// built-in middleware to handle urlencoded form data
app.use(express.urlencoded({ extended: false }));

// built-in middleware for json
app.use(express.json());

//middleware for cookies
app.use(cookieParser());

// routes
app.use('/register', require('./routes/register'));
app.use('/auth', require('./routes/auth'));
app.use('/refresh', require('./routes/refresh'));
app.use('/logout', require('./routes/logout'));
app.use(verifyJWT);
app.use('/manage-room', require('./routes/manageRoom'));
app.use('/university-exam', require('./routes/universityExam'));
app.use('/seat-allocation', require('./routes/seatAllocation'));

mongoose.connection.once('open', () => {
    console.log('Connected to MongoDB');
    app.listen(PORT, () => { console.log(`Server running on port ${PORT}...`) });
});
```

MongoDB

- Establishing a connection to the MongoDB server using a MongoDB client library mongoose for Node.js

- Performing CRUD operations on the MongoDB collections.

Schedule.js

C: > Users > alimo > OneDrive > Documents > GitHub > ESA-Backend > models > Schedule.js > ...

```
const mongoose = require('mongoose');

const ScheduleSchema = new mongoose.Schema({
    user: {
        type: String,
        required: true,
        ref: 'User'
    },
    sem: {
        type: Number,
        required: [true, 'Provide semester number'],
    },
    date: {
        type: Date,
        required: [true, 'Provide Date'],
        index: { expires: '100d' },
    },
    time: {
        type: String,
        required: [true, 'Provide Time'],

    },
    branch: {
        type: String,
        required: [true, 'Provide Branch'],
    },
    slot: {
        type: String,
        required: [true, 'Provide slot info'],
    },

    subcode: {
        type: String,
        required: [true, 'Provide subcode info'],
    }
},
    { timestamps: true }
);

module.exports = new mongoose.model('Schedule', ScheduleSchema);
```

openpyxl

- Python program that uses the openpyxl library to manipulate an Excel sheet.
- To execute the program using Node.js, we use the child_process module to spawn a Python process.

Conclusion

- The proposed exam hall seat allocation system aims to automate the seat allocation process, reduce administrative workload, and provide an efficient and effective solution that scales to accommodate large numbers of students.

- This project has the potential to improve the student experience and increase the security and accuracy of the exam process.

Future Scope

- It can be upgraded to allocate seats for internal exams
- An extra module to assign faculties for each examination halls can be added.
- An additional feature to send the allotted seat details to students via mail.

References

[1] Prof. Nilima Nikam, Akshata Jagdale, Gunjan Patil and Prachi Patil "Algorithm for efficient seat allocation process in college exam system", Yadavrao Tasgaonkar Institute Of Engineering And Technology.

[2] S. S. Ali and H. S. Al-Rawi "An Efficient Heuristic for Exam Seat Allocation", International Journal of Computer Science and Network Security (IJCSNS)

[3] Prof. S.Vasupongayya, W.Noodam, and P.Kongyong, “Developing Examination Management System: Senior Capstone Project, a Case Study”, World Academy of Science, Engineering and Technology, Vol:7 2013.

[4] Prof. Santa Kumar Chaki at ell “Algorithm For Efficient Seating Plan For Centralized Exam System” 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)

Snapshot

LOG IN

username

Enter your username

password

Enter your password

Sign In

Don't have an account? Sign Up

Snapshot

REGISTER

username :

email :

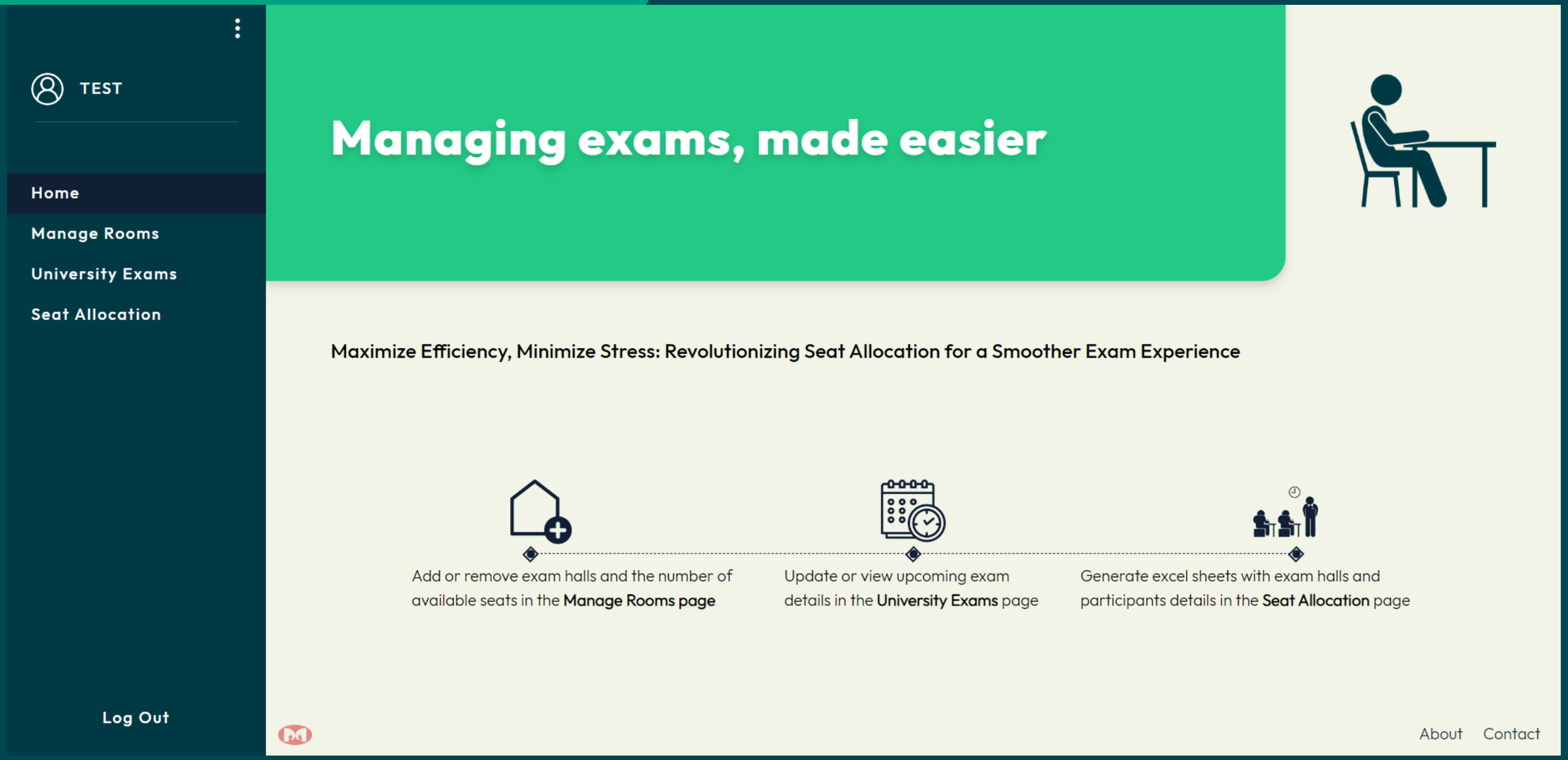
password :

confirm password :

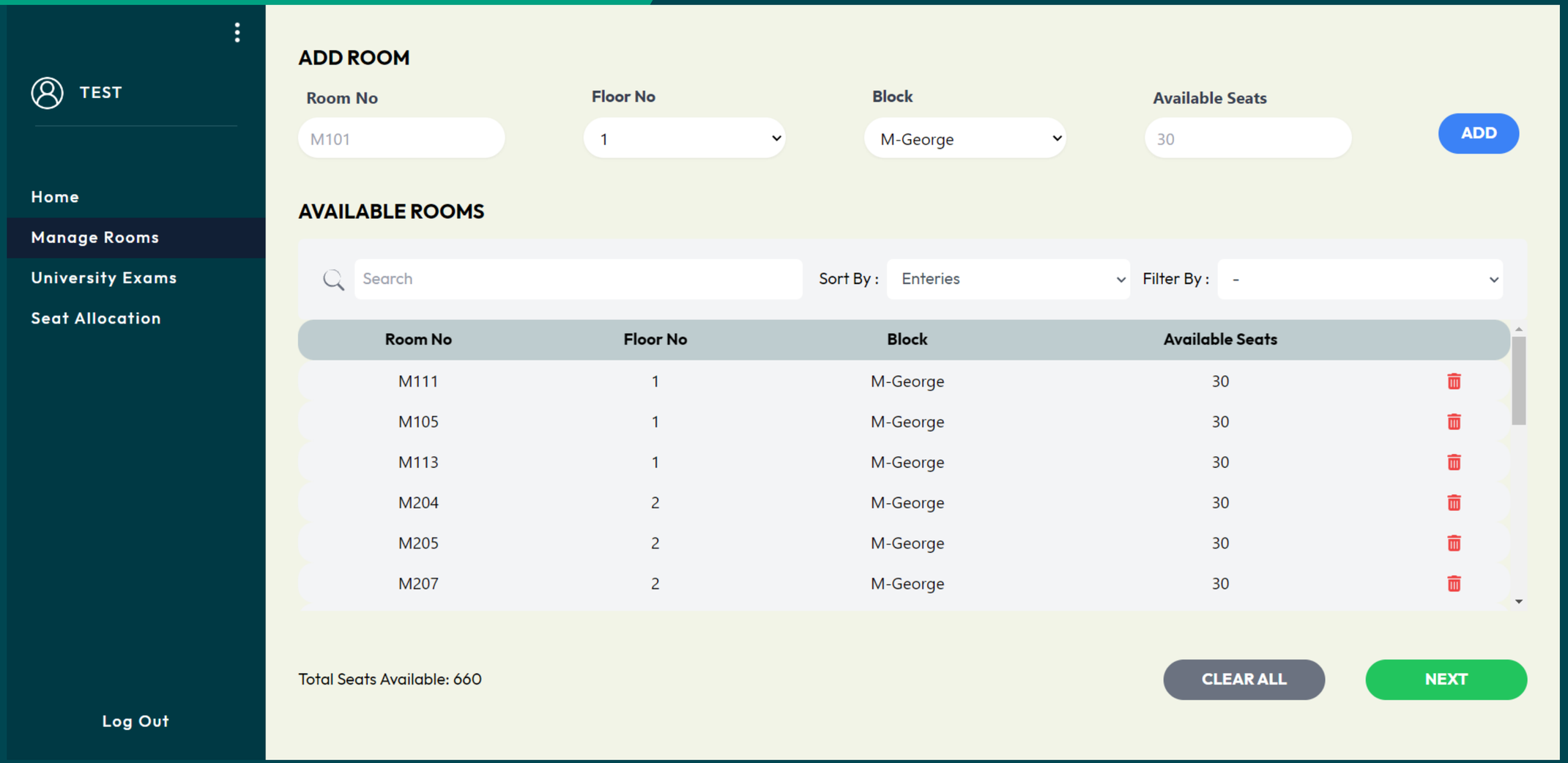
Sign Up

Don't have an account? Sign In

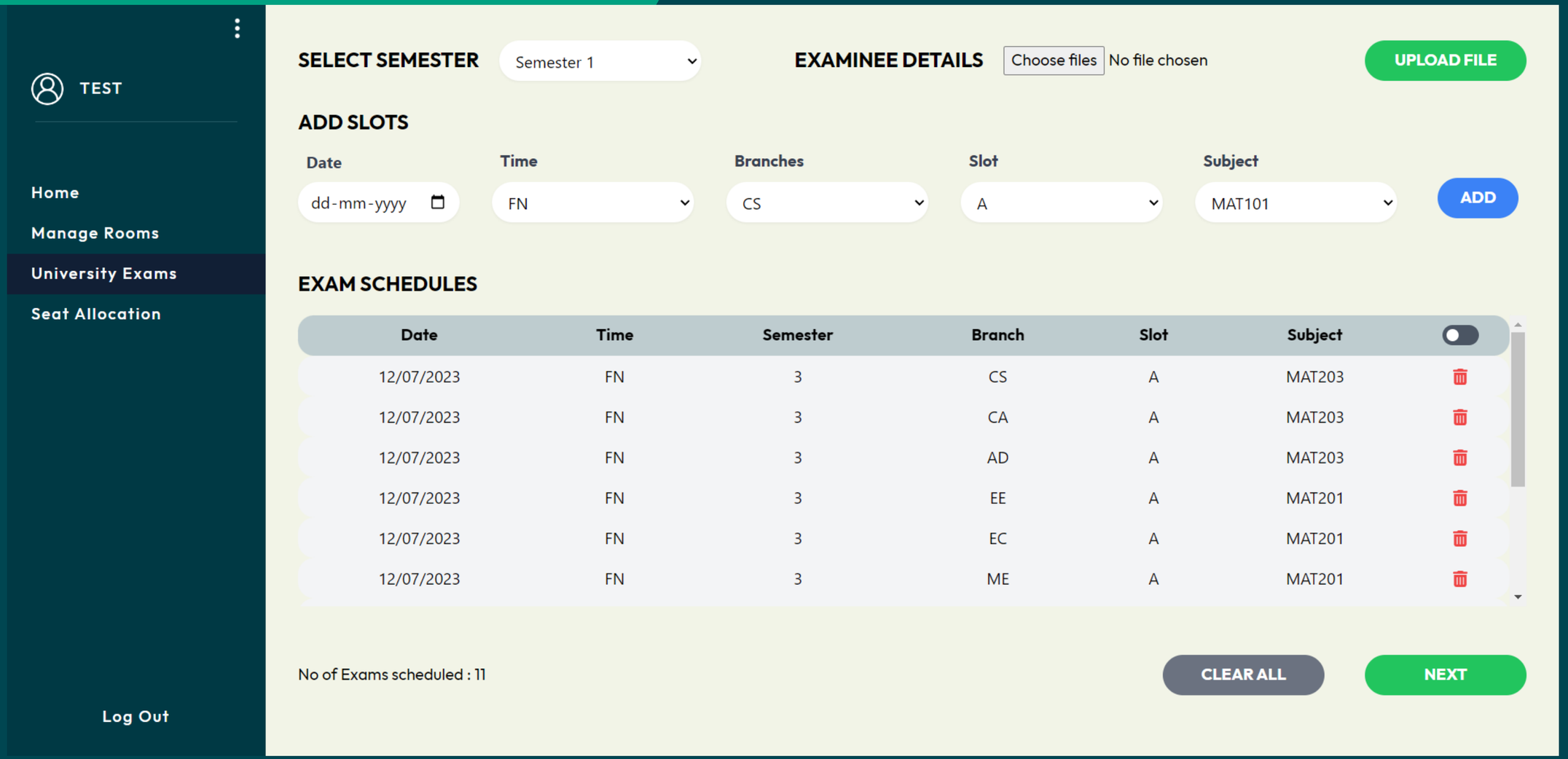
Snapshot


TEST
Home
Manage Rooms
University Exams
Seat Allocation
Log Out
Managing exams, made easier
Maximize Efficiency, Minimize Stress: Revolutionizing Seat Allocation for a Smoother Exam Experience
Add or remove exam halls and the number of available seats in the Manage Rooms page
Update or view upcoming exam details in the University Exams page
Generate excel sheets with exam halls and participants details in the Seat Allocation page
About
Contact

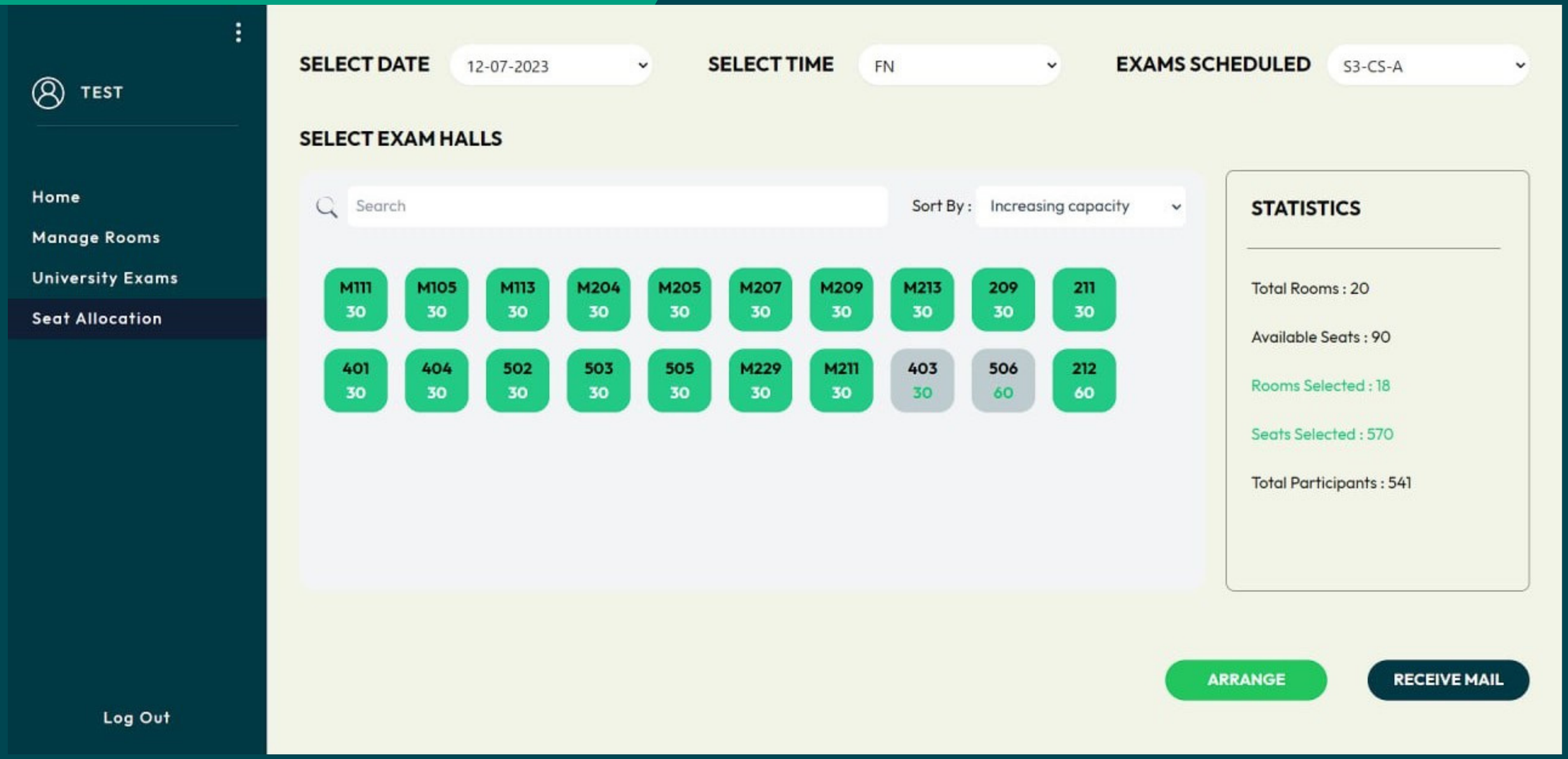
Snapshot


TEST
Home
Manage Rooms
University Exams
Seat Allocation
Log Out
ADD ROOM
Room No
M101
Floor No
1
Block
M-George
Available Seats
30
ADD
AVAILABLE ROOMS
Search
Sort By :
Enteries
Filter By :
-
Room No
Floor No
Block
Available Seats
M111 1 M-George 30
M105 1 M-George 30
M113 1 M-George 30
M204 2 M-George 30
M205 2 M-George 30
M207 2 M-George 30
Total Seats Available: 660
CLEAR ALL
NEXT

Snapshot


TEST
Home
Manage Rooms
University Exams
Seat Allocation
Log Out
SELECT SEMESTER
Semester 1
EXAMINEE DETAILS
Choose files
No file chosen
UPLOAD FILE
ADD SLOTS
Date
dd-mm-yyyy
Time
FN
Branches
CS
Slot
A
Subject
MAT101
ADD
EXAM SCHEDULES
Date | Time | Semester | Branch | Slot | Subject
12/07/2023 | FN | 3 | CS | A | MAT203
12/07/2023 | FN | 3 | CA | A | MAT203
12/07/2023 | FN | 3 | AD | A | MAT203
12/07/2023 | FN | 3 | EE | A | MAT201
12/07/2023 | FN | 3 | EC | A | MAT201
12/07/2023 | FN | 3 | ME | A | MAT201
No of Exams scheduled : 11
CLEAR ALL
NEXT

Snapshot


TEST
Home
Manage Rooms
University Exams
Seat Allocation
Log Out
SELECT DATE
12-07-2023
SELECT TIME
FN
EXAMS SCHEDULED
S3-CS-A
SELECT EXAM HALLS
Search
Sort By :
Increasing capacity
M111 30
M105 30
M113 30
M204 30
M205 30
M207 30
M209 30
M213 30
209 30
211 30
401 30
404 30
502 30
503 30
505 30
M229 30
M211 30
403 30
506 60
212 60
STATISTICS
Total Rooms : 20
Available Seats : 90
Rooms Selected : 18
Seats Selected : 570
Total Participants : 541
ARRANGE
RECEIVE MAIL

Thank You