**Revised Slide Breakdown**

1. **Title Slide**
   * **Project Name: photoStockage**
   * **Your Name**
   * **Date: February 20, 2025**
2. **Introduction**
   * **Brief self-introduction**
   * **Overview of the presentation**
3. **Project Overview**
   * **What is photoStockage?**
   * **Core functionalities**
   * **Target audience**
4. **Technical Specifications**
   * **Platforms**
   * **Technology stack**
   * **Integration**
5. **Backend Architecture**
   * **Overview of the backend structure**
   * **Code examples: Controllers, Routers, API endpoints**
   * **Security measures**
6. **Frontend Architecture**
   * **Overview of the frontend structure**
   * **Code examples: Components, State management, API integration**
   * **Responsive design**
7. **Data Management**
   * **Data models**
   * **Data storage**
   * **Backup strategy**
8. **Testing and Quality Assurance**
   * **Testing strategy**
   * **Code examples: Unit tests, Integration tests**
9. **Challenges and Solutions**
   * **Technical challenges faced**
   * **Solutions implemented**
10. **Project Timeline and Milestones**
    * **Phases**
    * **Challenges and solutions**
11. **Legal and Compliance**
    * **Intellectual property**
    * **Compliance**
12. **Conclusion and Approval**
    * **Summary**
    * **Additional considerations**
13. **Q&A**
    * **Open the floor for questions**

**Slide Content and Bullet Points**

**Slide 1: Title Slide**

* **Title: photoStockage**
* **Subtitle: A Platform for Free Photo Sharing and Community Engagement**
* **Your Name: [Your Name]**
* **Date: February 20, 2025**

**Slide 2: Introduction**

* **Self-Introduction:**
  + **Name**
  + **Background (e.g., education, experience)**
  + **Role in the project**
* **Presentation Overview:**
  + **Brief overview of what will be covered**
  + **Importance of the project**

**Slide 3: Project Overview**

* **What is photoStockage?**
  + **A platform for free photo sharing**
  + **Encourages community engagement**
* **Core Functionalities:**
  + **User registration**
  + **Photo uploading, downloading, and sharing**
  + **Liking and saving photos**
  + **Advanced searching and filtering**
* **Target Audience:**
  + **Photographers**
  + **Content creators**
  + **Graphic designers**
  + **Web developers**
  + **Marketing professionals**
  + **Students and educators**

**Slide 4: Technical Specifications**

* **Platforms:**
  + **Platform-agnostic**
  + **Responsive design**
* **Technology Stack:**
  + **Front-end: Next.js**
  + **Back-end: Node.js, Express.js**
  + **Database: PostgreSQL**
* **Integration:**
  + **Third-party services for security**
  + **Large Language Models for assistance**

**Slide 5: Backend Architecture**

* **Overview of the Backend Structure:**
  + **MVC architecture (without the view part)**
  + **Code examples:**
    - **Controllers: Handling business logic**
    - **Routers: Defining API endpoints**
    - **API endpoints: Examples of CRUD operations**
* **Security Measures:**
  + **Authentication and authorization**
  + **Input sanitization**
  + **Rate limiting**

**Slide 6: Frontend Architecture**

* **Overview of the Frontend Structure:**
  + **Component-based architecture**
  + **Code examples:**
    - **Components: Reusable UI elements**
    - **State management: Using React's useState and useEffect**
    - **API integration: Fetching data from the backend**
* **Responsive Design:**
  + **Mobile-first approach**
  + **Use of Tailwind CSS for styling**

**Slide 7: Data Management**

* **Data Models:**
  + **Seven tables: users, photos, likes, categories, photos\_categories, comments, downloads**
* **Data Storage:**
  + **Hosted on Namecheap**
  + **Daily, weekly, and monthly backups**
* **Backup Strategy:**
  + **Automated backups**
  + **Regular local backups**

**Slide 8: Testing and Quality Assurance**

* **Testing Strategy:**
  + **Comprehensive unit testing**
  + **Backend and frontend tests**
  + **Code examples:**
    - **Unit tests: Testing individual components**
    - **Integration tests: Testing API endpoints**
* **Successful Test Results:**
  + **Ensuring the platform meets quality standards**

**Slide 9: Challenges and Solutions**

* **Technical Challenges Faced:**
  + **Prototyping with Adobe XD**
  + **Reconversion of site style for mobile-first design**
  + **Management of JWT token and HTTP-only cookies**
* **Solutions Implemented:**
  + **Watched tutorials and sought guidance from forums**
  + **Re-read styles and re-imagined them as mobile-first**
  + **Learned from training projects and implemented JWT and cookie management**

**Slide 10: Project Timeline and Milestones**

* **Phases:**
  + **Specification document creation (May 2024)**
  + **UML diagrams and visual definitions (June-July 2024)**
  + **Backend development (August-September 2024)**
  + **Frontend development (September-December 2024)**
  + **Finalization (February 2025)**
* **Challenges and Solutions:**
  + **Prototyping with Adobe XD**
  + **Reconversion of site style for mobile-first design**
  + **Management of JWT token and HTTP-only cookies**

**Slide 11: Legal and Compliance**

* **Intellectual Property:**
  + **Licensed under the MIT license**
  + **Open-source code**
* **Compliance:**
  + **GDPR and European laws**
  + **Terms and conditions**

**Slide 12: Conclusion and Approval**

* **Summary:**
  + **Recap of key points**
  + **Importance of the project**
* **Additional Considerations:**
  + **Documentation**
  + **Future improvements**

**Slide 13: Q&A**

* **Open the Floor for Questions:**
  + **Encourage the audience to ask questions**
  + **Provide contact information for follow-up**

**Script to Memorize**

**Introduction**

**"Good [morning/afternoon], everyone. I am [Your Name], and I am excited to present to you today my project, photoStockage. I have a background in [your background], and I have been passionate about [your passion related to the project]. Today, I will walk you through the development, objectives, technical specifications, and future plans for photoStockage. Let's get started."**

**Project Overview**

**"photoStockage is a platform I designed to facilitate the free sharing of high-quality photographs. It allows users to upload, download, and share photos, fostering a community of photographers and content creators. The core functionalities include user registration, photo management, and advanced searching and filtering options. Our target audience ranges from photographers and content creators to graphic designers, web developers, marketing professionals, students, and educators."**

**Technical Specifications**

**"I designed photoStockage to be platform-agnostic with a responsive design. The technology stack includes Next.js for the front-end, Node.js and Express.js for the back-end, and PostgreSQL for the database. I integrated third-party services for security and leveraged Large Language Models for assistance."**

**Backend Architecture**

**"The backend follows an MVC architecture without the view part. I handled business logic in controllers, defined API endpoints in routers, and implemented security measures such as authentication, authorization, input sanitization, and rate limiting. Here are some code examples:"**

* **Controllers:**
* **// Example of a controller function**
* **const getPhotos = async (req, res) => {**
* **try {**
* **const photos = await Photo.findAll();**
* **res.status(200).json(photos);**
* **} catch (error) {**
* **res.status(500).json({ error: 'Internal Server Error' });**
* **}**

**};**

* **Routers:**
* **// Example of a router definition**
* **const photoRouter = express.Router();**

**photoRouter.get('/', getPhotos);**

* **API Endpoints:**
* **// Example of an API endpoint**

**app.use('/api/photos', photoRouter);**

**Frontend Architecture**

**"The frontend follows a component-based architecture using Next.js. I created reusable UI components, managed state using React's useState and useEffect, and integrated with the backend API. Here are some code examples:"**

* **Components:**
* **// Example of a reusable component**
* **const PhotoCard = ({ photo }) => {**
* **return (**
* **<div className="photo-card">**
* **<img src={photo.url} alt={photo.title} />**
* **<h3>{photo.title}</h3>**
* **</div>**
* **);**

**};**

* **State Management:**
* **// Example of state management**
* **const [photos, setPhotos] = useState([]);**
* **useEffect(() => {**
* **fetch('/api/photos')**
* **.then(response => response.json())**
* **.then(data => setPhotos(data));**

**}, []);**

* **API Integration:**
* **// Example of API integration**
* **const fetchPhotos = async () => {**
* **const response = await fetch('/api/photos');**
* **const data = await response.json();**
* **setPhotos(data);**

**};**

**Data Management**

**"My data management strategy includes seven tables: users, photos, likes, categories, photos\_categories, comments, and downloads. Data is stored on Namecheap with daily, weekly, and monthly backups to ensure data safety and integrity."**

**Testing and Quality Assurance**

**"I implemented comprehensive unit testing for both the back-end and front-end components. All tests have successfully passed, demonstrating that the platform meets the desired quality standards. Here are some code examples:"**

* **Unit Tests:**
* **// Example of a unit test**
* **test('should fetch photos', async () => {**
* **const response = await fetch('/api/photos');**
* **const data = await response.json();**
* **expect(data.length).toBeGreaterThan(0);**

**});**

* **Integration Tests:**
* **// Example of an integration test**
* **test('should create a new photo', async () => {**
* **const response = await fetch('/api/photos', {**
* **method: 'POST',**
* **body: JSON.stringify({ title: 'New Photo', url: 'http://example.com/photo.jpg' }),**
* **headers: { 'Content-Type': 'application/json' }**
* **});**
* **expect(response.status).toBe(201);**

**});**

**Challenges and Solutions**

**"During the development, I faced several technical challenges. Prototyping with Adobe XD was cumbersome due to my lack of experience in graphic design. I overcame this by watching tutorials and seeking guidance from forums. Another challenge was reconverting the site style for a mobile-first design. I solved this by re-reading the styles and re-imagining them as mobile-first. The biggest challenge was managing JWT tokens and HTTP-only cookies. I learned from training projects and implemented JWT and cookie management successfully."**

**Project Timeline and Milestones**

**"The project was divided into manageable phases. I started with the specification document in May 2024, followed by UML diagrams and visual definitions. Backend development began in August 2024, and frontend development started in September 2024. The project was completed in December 2024, with finalization in February 2025. Some challenges I faced included prototyping with Adobe XD, reconverting the site style for mobile-first design, and managing JWT tokens and HTTP-only cookies."**

**Legal and Compliance**

**"I licensed photoStockage under the MIT license, making the code open-source. I ensured compliance with GDPR and European laws, ensuring user data protection and privacy. Our terms and conditions outline user responsibilities and platform usage guidelines."**

**Conclusion and Approval**

**"In conclusion, photoStockage brings a new resource to developers and content creators, providing a free-of-charge service that is fast, easy to use, and available on all devices. It fosters a sense of community by enabling user communication. Thank you for your attention. I am now open to any questions you may have."**