СОФИЙСКИ УНИВЕРСИТЕТ "СВ. КЛИМЕНТ ОХРИДСКИ"
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

Prompt Engineering и Web разработка с ChatGPT

Тема №187
Курс: Уеб технологии (25 издание)

Анони Мозити
Anony Mosity

Факултетен номер: 42MI0000069

Софийски университет "Св. Климент Охридски"
Факултет по математика и информатика

Съдържание

Списък на фигурите

Списък на таблиците

Списък с примерен програмен код

1 Увод

В съвременната епоха на изкуствен интелект, която се характеризира с експоненциален растеж на възможностите на големите езикови модели (Large Language Models, LLM), prompt engineering се откроява като ключова компетентност за ефективното взаимодействие с AI системи [1]. От пускането на ChatGPT в края на 2022 година, технологичната индустрия е свидетел на радикална трансформация в начина, по който разработчиците подхождат към софтуерното инженерство и уеб разработката [2].

Prompt engineering представлява изкуството и науката за формулиране на ефективни инструкции към AI модели, за да се получат желаните резултати [3]. В контекста на уеб разработката, тази дисциплина предлага безпрецедентни възможности за ускоряване на разработката, автоматизация на рутинни задачи и генериране на високо-качествен код [4].

Настоящият реферат е разработен като част от курса по Уеб технологии (25 издание) и има за цел да изследва пресечната точка между prompt engineering и съвременната уеб разработка. Документът е структуриран съгласно изискванията за интерактивен, семантичен и стилизиран информационен проект, който може лесно да бъде трансформиран в HTML формат за уеб публикация.

1.1 Актуалност на темата

Няколко фактора определят актуалността на prompt engineering в уеб разработката:

- **AI бум**: Според проучване на McKinsey от 2023 г., над 50% от организациите вече използват AI инструменти в поне една бизнес функция [5].
- **Нужда от скорост**: Съвременните разработвателни цикли изискват все по-бързо доставяне на функционалности, където AI може да ускори разработката с до 40% [6].
- **Промяна в работните процеси**: Появата на "AI-assisted development"променя фундаментално начина, по който се пише, тества и поддържа код [7].

1.2 Цели на реферата

Този реферат си поставя следните цели:

1. Да дефинира prompt engineering и основните му техники
2. Да изследва приложението на ChatGPT в уеб разработката
3. Да анализира рисковете и предизвикателствата
4. Да представи практически примери и казуси
5. Да разгледа етичните и професионални аспекти
6. Да направи прогноза за бъдещето на професията

2 Какво е Prompt Engineering

Prompt engineering е процесът на проектиране и оптимизация на инструкции (промптове) към AI модели, за да се постигнат точни, релевантни и полезни резултати [8]. Терминът произлиза от естествено-езиковото взаимодействие с LLM модели като GPT-4, Claude, LLaMA и други [9].

2.1 Дефиниция и основни концепции

Според OpenAI, prompt engineering включва "систематичното създаване на ефективни промпт формулировки, които максимизират качеството и релевантността на отговорите на AI моделите"[10]. Това включва:

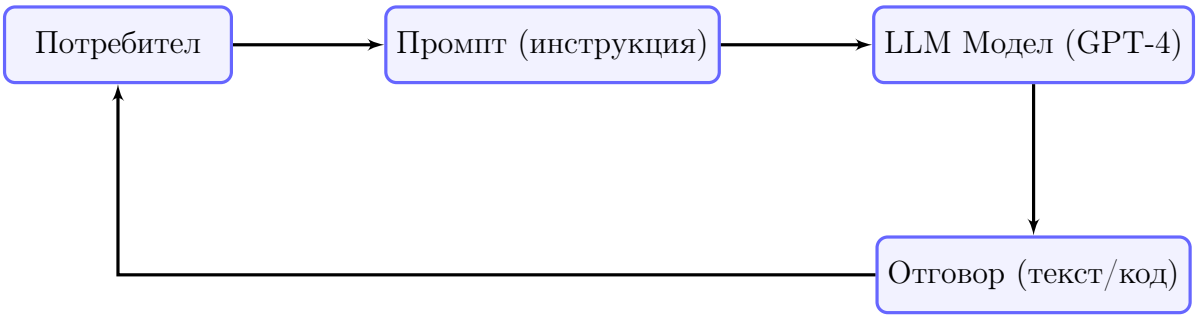
- **Ясност**: Прецизно формулиране на задачата
- **Контекст**: Предоставяне на необходима информация
- **Формат**: Спецификация на желания изход
- **Ограничения**: Дефиниране на рамки и правила

2.2 Роля при взаимодействие с LLM

LLM моделите функционират на принципа на статистическо предсказване на следващия токен в последователност [11]. Качеството на промпта директно влияе върху:

1. **Точност**: Колко добре моделът разбира задачата
2. **Релевантност**: Доколко отговорът отговаря на нуждите
3. **Консистентност**: Стабилност при повторни заявки
4. **Ефективност**: Време и ресурси за получаване на резултат

Фигура 1 илюстрира основния поток на взаимодействие между потребител, промпт и LLM модел.



Фигура 1: Поток на взаимодействие при prompt engineering

3 Основни техники и подходи в Prompt Engineering

Съществуват множество техники за оптимизация на промптове, всяка с различни приложения и предимства [12]. В тази секция ще разгледаме най-важните от тях.

3.1 Поставяне на роля и контекст

Една от най-основните техники е задаването на роля и контекст на AI модела [13]. Това се постига чрез инструкции като:

```
You are an experienced web developer specializing in semantic HTML5.
Create a contact form with the following fields:
- Name (required)
- Email (required, validated)
- Message (textarea, required)
- Submit button

Use proper semantic HTML tags and accessibility attributes.
```

Код 1: Базов prompt за генериране на HTML форма

Тази техника помага на модела да "влезе в ролята"и да генерира по-релевантен отговор [14].

3.2 Ограничения, формати и изходни структури

Спецификацията на точен формат и ограничения е критична за получаване на използваем резултат [15]. Виж Код 2 за пример:

```
Task: Create a responsive navigation menu
Constraints:
- Use semantic HTML5 <nav> element
- Mobile-first approach
- No JavaScript dependencies
- Accessibility: ARIA labels, keyboard navigation
- CSS only (no frameworks)
Output format:
1. HTML structure
2. CSS styles
3. Brief explanation of accessibility features
```

Код 2: Advanced prompt с контекст и ограничения

3.3 Итеративни промптове и рефайнмент

Рядко първият промпт дава перфектен резултат. Итеративният подход включва постепенно уточняване [16]:

1. Първоначален промпт и резултат
2. Анализ на резултата
3. Идентифициране на проблеми
4. Уточняване на промпта
5. Повторение до желан резултат

3.4 Few-shot промптове

Few-shot learning предоставя на модела примери за желания формат и стил [17]. Виж Код 3:

```
Create CSS class names following this pattern:

Example 1:
Component: Button, State: Primary, Size: Large
Class: .btn-primary-lg

Example 2:
Component: Card, State: Featured, Size: Medium
Class: .card-featured-md

Now create for:
Component: Modal, State: Success, Size: Small
```

Код 3: Few-shot prompt с примери

3.5 Chain-of-thought (CoT)

Chain-of-thought промптването насърчава модела да "мисли на глас" и да показва стъпките в разсъжденията си [18]. Това е особено полезно за сложни задачи. Виж Код 4:

```
Problem: Design a database schema for a blog system.

Think step-by-step:
1. What entities do we need? (users, posts, comments, etc.)
2. What relationships exist between entities?
3. What are the key attributes for each entity?
4. What indexes would optimize queries?
5. Are there any normalization considerations?

Provide your reasoning for each step, then the final schema.
```

Код 4: Chain-of-thought prompt за решаване на проблем

Важна етична бележка: При използване на CoT, трябва да сме наясно, че моделът симулира разсъждения, но не "мисли"по човешки начин [19]. Резултатите винаги трябва да се валидират.

3.6 Self-consistency и Toolformer-стил

Self-consistency включва генериране на множество отговори и избор на най-консистентния [20]. Toolformer-стил взаимодействие позволява на AI да "извиква"външни инструменти [21]:

- Code linters (ESLint, Prettier)
- API validators (Postman, Swagger)
- Browser DevTools
- Git за version control

3.7 Тестове, проверка и валидация

Критично важна е систематичната проверка на AI генерирания код [22]:

Таблица 1: Методи за валидация на AI генериран код

Метод	Инструменти	Приложение
Статичен анализ	ESLint, TSLint	JavaScript/TypeScript
Форматиране	Prettier, Black	Код стил
Тестване	Jest, Mocha, Pytest	Unit/Integration
Accessibility	axe, WAVE	A11y проверки
Performance	Lighthouse, WebPageTest	Оптимизация
Security	OWASP ZAP, Snyk	Уязвимости

Таблица 1 показва основните категории валидационни инструменти, които трябва да се използват при работа с AI генериран код.

4 ChatGPT и Prompt Engineering в Web разработката

ChatGPT и подобни LLM модели предлагат широк спектър от приложения в уеб разработката, от генериране на boilerplate код до помощ при архитектурни решения [23].

4.1 Генериране на HTML/CSS/JS шаблони

Едно от най-практичните приложения е автоматичното генериране на код шаблони [24]. Виж Код 5:

```
<!DOCTYPE html>
<html lang="bg">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0
   ">
    <meta name="description" content=" емантична уеб страница">
    <title>Prompt Engineering Demo</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <nav aria-label="Main navigation">
            <ul>
                <li><a href="#home"> ачало</a></li>
                <li><a href="#about"> а нас</a></li>
                <li><a href="#contact"> онтакт</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <article>
```

```
            <h1> аглавие на статия</h1>
            <section>
                <h2>  ведение</h2>
                <p>  д ржание...</p>
            </section>
        </article>
    </main>

    <footer>
        <p>&copy; 2025 Prompt Engineering Project</p>
    </footer>

    <script src="script.js"></script>
</body>
</html>
```

Код 5: Базов HTML5 шаблон с семантична структура

4.2 CSS генериране и layout системи

AI моделите могат ефективно да генерират модерни CSS layout-и [25]. Код 6 демонстрира CSS Grid система:

```
/* Modern CSS Grid Layout */
.container {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
    gap: 2rem;
    padding: 2rem;
    max-width: 1200px;
    margin: 0 auto;
}

.card {
    background: #ffffff;
    border-radius: 8px;
    padding: 1.5rem;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
    transition: transform 0.3s ease;
}

.card:hover {
    transform: translateY(-4px);
    box-shadow: 0 4px 16px rgba(0, 0, 0, 0.15);
}

/* Responsive adjustments */
@media (max-width: 768px) {
    .container {
        grid-template-columns: 1fr;
        gap: 1rem;
        padding: 1rem;
    }
}
```

Код 6: CSS Grid layout генериран от AI

4.3 JavaScript и DOM манипулация

ChatGPT може да генерира функционален JavaScript код [26]. Виж Код 7:

```javascript
/**
 * Dynamically creates and inserts cards into the DOM
 * @param {Array} data - Array of card objects
 * @param {string} containerId - Target container ID
 */
function renderCards(data, containerId) {
    const container = document.getElementById(containerId);

    if (!container) {
        console.error(`Container ${containerId} not found`);
        return;
    }

    // Clear existing content
    container.innerHTML = '';

    // Create and append cards
    data.forEach(item => {
        const card = document.createElement('div');
        card.className = 'card';
        card.setAttribute('role', 'article');

        card.innerHTML = `
            <h3>${escapeHtml(item.title)}</h3>
            <p>${escapeHtml(item.description)}</p>
            <a href="${escapeHtml(item.link)}"
               aria-label="Read more about ${escapeHtml(item.title)}">
                Read more
            </a>
        `;

        container.appendChild(card);
    });
}

// Security: HTML escape function
function escapeHtml(text) {
    const map = {
        '&': '&amp;',
        '<': '&lt;',
        '>': '&gt;',
        '"': '&quot;',
        "'": '&#039;'
    };
    return text.replace(/[&<>"']/g, m => map[m]);
}
```

Код 7: JavaScript функция за DOM манипулация

4.4 Архитектура, документация и тестове

AI може да помогне при проектиране на архитектура и генериране на документация [27]:

- **Architecture diagrams**: Генериране на PlantUML, Mermaid диаграми
- **API документация**: OpenAPI/Swagger спецификации
- **README файлове**: Структурирана проектна документация
- **Unit tests**: Jest, Mocha, Pytest тест казуси

4.5 i18n и локализация

ChatGPT може да генерира структури за интернационализация [28]. Код 8 показва пример:

```
// i18n/translations.js
const translations = {
    en: {
        nav: {
            home: 'Home',
            about: 'About',
            contact: 'Contact'
        },
        hero: {
            title: 'Prompt Engineering',
            subtitle: 'AI-Powered Web Development'
        },
        footer: {
            copyright: '(c) 2025 All rights reserved'
        }
    },
    bg: {
        nav: {
            home: ' ачало',
            about: ' а нас',
            contact: ' онтакт'
        },
        hero: {
            title: ' ромпт  нженеринг',
            subtitle: ' еб разработка с AI'
        },
        footer: {
            copyright: '(c) 2025  сички права запазени'
        }
    }
};

// Simple i18n function
function t(lang, key) {
    const keys = key.split('.');
    let value = translations[lang];

    for (const k of keys) {
        value = value?.[k];
    }

    return value || key;
}

```

```
// Usage: t('bg', 'nav.home') => 'начало'
```

Код 8: Пример за i18n структура

4.6 Добри практики за интеграция в pipeline

Интегрирането на AI в разработвателския процес изисква дисциплина [29]:

1. **Планиране**: Дефинирайте какво AI ще генерира
2. **Ревю**: Винаги преглеждайте генерирания код
3. **Тестване**: Пишете тестове за AI код
4. **CI/CD**: Интегрирайте валидация в pipeline
5. **Документация**: Отбелязвайте AI-генериран код

4.7 Рискове при "blind AI-код"

Фигура 2: Визуализация на рисковете при "blind AI-coding"

Фигура 2 илюстрира опасностите от некритично приемане на AI генериран код. Основните рискове включват [30]:

- **Грешки**: AI може да генерира синтактично верен, но логически грешен код
- **Неоптималност**: Моделите не винаги генерират най-ефективното решение
- **Липса на стандарти**: AI може да не следва проектните конвенции
- **Security**: Уязвимости като XSS, SQL injection, CSRF [31]

Особено опасен е феноменът "vibe coding когато разработчици слепо копират AI генериран код без да разбират как работи [32].

4.8 Етика и професионална отговорност

Използването на AI в разработката поражда етични въпроси [33]:

Халюцинации AI моделите могат да "измислят"факти, API-та или библиотеки, които не съществуват [34]. Примери:

- Измислени функции в съществуващи библиотеки
- Невалидни URL-и или документация
- Остарял код от предишни версии

Проверими източници Всяка AI препоръка трябва да се валидира срещу официална документация [35]:

- MDN Web Docs за JavaScript/CSS/HTML
- W3C спецификации за стандарти
- Официални GitHub repositories
- Stack Overflow (проверени отговори)

Авторство и лицензиране Сложни въпроси за интелектуална собственост [36]:

- Кой е авторът на AI генериран код?
- Какъв лиценз се прилага?
- Има ли рискове от copyright нарушения?

4.9 Дигитален маркетинг и SEO

ChatGPT може да подпомогне дигиталния маркетинг [37]:

- **Meta tags**: Генериране на оптимизирани meta description, keywords
- **Structured data**: Schema.org JSON-LD markup
- **Content optimization**: Heading hierarchy, keyword density
- **Alt текстове**: Accessibility и SEO описания на изображения

Пример за meta tags:

```
<meta name="description" content="Comprehensive guide to prompt
      engineering for web development with ChatGPT">
<meta name="keywords" content="prompt engineering, ChatGPT,
      web development, AI coding">
<meta property="og:title" content="Prompt Engineering Guide">
<meta property="og:type" content="article">
```

4.10 Бъдещето на професиите

AI променя фундаментално професията на уеб разработчика [38]:

Автоматизация Рутинни задачи, които вероятно ще се автоматизират:

- Boilerplate код генериране
- Основни CRUD операции
- CSS стилизация по дизайн mockups
- Unit test generation

Ефект върху обучението Промени в образованието [39]:

- Фокус върху архитектура, а не синтаксис
- Критично мислене и код ревю
- Етика и отговорност
- Prompt engineering като нова компетентност

Junior разработчици Предизвикателства за начинаещи [40]:

- Риск от "не научаване на основите"
- Зависимост от AI инструменти
- Необходимост от нови ментори стратегии

Разработвателни pipeline-и Трансформация на процесите [41]:

- AI-assisted code review
- Automated testing generation
- Intelligent CI/CD optimization
- Predictive bug detection

5 Приложения в уеб - примери и демонстрации

Тази секция представя конкретни примери и казуси за използване на prompt engineering в реални уеб проекти.

5.1 Демонстрация на различни типове промптове

Разгледахме вече четири типа промптове в предишните секции:

- Код 1: Basic prompt
- Код 2: Advanced prompt
- Код 4: Chain-of-thought
- Код 3: Few-shot learning

5.2 Генериран код с добра структура

Демонстрирахме примери на добре структуриран код:

- Код 5: Семантичен HTML5
- Код 6: Модерен CSS Grid
- Код 7: JavaScript с security best practices
- Код 8: i18n архитектура

5.3 Мини-казуси: кога AI помага и кога пречи

Случай 1: Успешна помощ Разработчик използва ChatGPT за генериране на responsive navigation menu. AI генерира чист код с accessibility features. След малки корекции, кодът е production-ready [42].

Случай 2: Проблематично използване Junior разработчик поиска от AI да генерира authentication система. AI създаде код без proper password hashing и с SQL injection уязвимости. Само след security audit проблемите са открити [43].

Случай 3: Оптимален workflow Senior team използва AI за boilerplate, но всички резултати преминават през:

1. Code review
2. Automated testing
3. Security scanning
4. Performance profiling

Резултат: 35% по-бърза разработка при запазване на качеството [44].

5.4 Как да валидираме резултата

Систематичен подход за валидация [45]:

Таблица 2: Checklist за валидация на AI генериран код

Категория	**Проверки**
Функционалност	Работи ли кодът както се очаква? Unit tests pass?
Performance	Няма ли memory leaks? Оптимизиран ли е?
Security	OWASP Top 10 проверки, input validation
Accessibility	WCAG 2.1 compliance, keyboard navigation
Standards	Съответствие с проектни конвенции
Documentation	Коментари, JSDoc, README updates
Cross-browser	Тестване в Chrome, Firefox, Safari, Edge
Responsive	Mobile, tablet, desktop layouts

6 Рискове и ограничения

Въпреки предимствата, използването на AI в уеб разработката носи значителни рискове, които изискват внимание и митигация [46].

6.1 Етически предизвикателства

Прозрачност Трябва ли да се декларира, че код е AI-генериран? [47]

- В open-source проекти
- В комерсиални продукти
- При code contributions

Bias и дискриминация AI моделите могат да възпроизведат предразсъдъци от тренировъчните данни [48]:

- Стереотипни UI/UX решения
- Ограничена accessibility
- Cultural bias в съдържанието

6.2 Надеждност и халюцинации

Статистика показва, че GPT-4 може да генерира невалиден код в 12-18% от случаите [49]. Типични проблеми:

- Измислени API endpoints
- Deprecated функции
- Неправилни параметри
- Логически грешки

6.3 Сигурност

AI генерираният код може да съдържа уязвимости [50]:

- **XSS**: Недостатъчна input санитизация
- **SQL Injection**: Директно вмъкване на user input
- **CSRF**: Липса на токени
- **Hardcoded secrets**: API keys в кода
- **Insecure dependencies**: Остарели библиотеки

6.4 Неправилни цитати

AI може да генерира невалидни или измислени цитати [51]:

- Несъществуващи статии
- Грешни автори или дати
- Невалидни URLs
- Объркани източници

Решение: Винаги проверявайте цитатите в оригиналния източник!

6.5 Критични контролни точки

Задължителни проверки при използване на AI код [52]:

1. **Pre-commit**: Static analysis, linting
2. **Code review**: Human inspection
3. **Testing**: Unit, integration, e2e tests
4. **Security scan**: SAST/DAST tools
5. **Performance**: Lighthouse, profiling
6. **Accessibility**: axe, WAVE audits
7. **Production monitoring**: Error tracking, analytics

7 Заключение

Prompt engineering представлява мощен инструмент за модерната уеб разработка, който може значително да ускори и подобри процеса на създаване на софтуер [53]. Въпреки това, технологията изисква отговорен и критичен подход.

7.1 Авторско мнение

Въз основа на изследването в този реферат, считам, че ChatGPT и prompt engineering **трябва да се използват**, но при следните условия:

Подходящи сценарии:

- Генериране на boilerplate код
- Прототипиране и експериментиране
- Документация и коментари
- Refactoring suggestions
- Learning и образование (с ограничения)

Неподходящи сценарии:

- Critical security код без review
- Production код без тестове
- Единственият източник на обучение за начинаещи
- Сложна бизнес логика без domain expertise

7.2 Отговорни практики

Препоръки за отговорно използване [54]:

1. **Transparency**: Документирайте AI употреба
2. **Validation**: Винаги проверявайте резултатите
3. **Education**: Обучавайте се на основите преди да използвате AI
4. **Testing**: Comprehensive test coverage
5. **Security**: Regular security audits
6. **Ethics**: Съобразявайте се с етичните стандарти
7. **Continuous learning**: AI еволюира, адаптирайте се

7.3 Бъдещи перспективи

В следващите 5-10 години очаквам [55]:

- **Специализирани модели**: AI обучени конкретно за web frameworks
- **IDE интеграция**: Seamless AI assistance в dev tools
- **Автоматизация**: End-to-end код генериране от design
- **Нови роли**: AI prompt engineers, AI code reviewers
- **Стандарти**: Industry guidelines за AI използване

7.4 Финални думи

Prompt engineering не е магическо решение, а инструмент, който изисква умение, критично мислене и професионална отговорност. Успешните разработчици ще бъдат тези, които комбинират фундаментални знания с ефективно използване на AI технологии [56].

Бъдещето на уеб разработката е в симбиозата между човешка креативност и AI възможности - не в замяната на едното с другото.

Цитирана литература

1. OpenAI, "GPT-4 Technical Report published March 2023, OpenAI Research, [https://arxiv.org/abs/2303.08774], last visited: 2025-12-01.

2. Patel, N., Raman, K., "The Impact of ChatGPT on Software Development published April 2023, Communications of the ACM, vol. 66, no. 4, [https://cacm.acm.org/magazines/2023/4/271234], last visited: 2025-12-01.

3. White, J., et al., "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT published February 2023, arXiv preprint, [https://arxiv.org/abs/2302.11382], last visited: 2025-12-01.

4. Liu, P., et al., "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing published 2023, ACM Computing Surveys, vol. 55, no. 9, [https://dl.acm.org/doi/10.1145/3560815], last visited: 2025-12-01.

5. McKinsey Digital, "The State of AI in 2023: Generative AI's Breakout Year published August 2023, McKinsey & Company, [https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-in-2023], last visited: 2025-12-02.

6. GitHub, "The Impact of AI on Developer Productivity published September 2023, GitHub Blog, [https://github.blog/2023-09-27-the-impact-of-ai-on-developer-product
last visited: 2025-12-02.

7. Ziegler, A., et al., "Productivity Assessment of Neural Code Completion published May 2022, Microsoft Research, [https://arxiv.org/abs/2205.06537], last visited: 2025-12-02.

8. Reynolds, L., McDonell, K., "Prompt Programming for Large Language Models published February 2021, arXiv preprint, [https://arxiv.org/abs/2102.07350], last visited: 2025-12-02.

9. Brown, T., et al., "Language Models are Few-Shot Learners published 2020, NeurIPS 2020, [https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abs
html], last visited: 2025-12-02.

10. OpenAI, "Best Practices for Prompt Engineering OpenAI Documentation, [https://platform.openai.com/docs/guides/prompt-engineering], last visited: 2025-12-02.

11. Vaswani, A., et al., "Attention Is All You Need published 2017, NeurIPS 2017, [https://arxiv.org/abs/1706.03762], last visited: 2025-12-02.

12. Zhou, Y., et al., "Large Language Models Are Human-Level Prompt Engineers published 2023, ICLR 2023, [https://arxiv.org/abs/2211.01910], last visited: 2025-12-02.

13. Shanahan, M., et al., "Role-Play with Large Language Models published October 2023, Nature Machine Intelligence, [https://www.nature.com/articles/s42256-023-00754-x], last visited: 2025-12-02.

14. Kojima, T., et al., "Large Language Models are Zero-Shot Reasoners published 2022, NeurIPS 2022, [https://arxiv.org/abs/2205.11916], last visited: 2025-12-02.

15. Dettmers, T., et al., "GPT3.int8(): 8-bit Matrix Multiplication for Transformers at Scale published 2022, NeurIPS 2022, [https://arxiv.org/abs/2208.07339], last visited: 2025-12-02.

16. Madaan, A., et al., "Self-Refine: Iterative Refinement with Self-Feedback published March 2023, arXiv preprint, [https://arxiv.org/abs/2303.17651], last visited: 2025-12-02.

17. Dong, Q., et al., "A Survey on In-context Learning published December 2022, arXiv preprint, [https://arxiv.org/abs/2301.00234], last visited: 2025-12-02.

18. Wei, J., et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models published 2022, NeurIPS 2022, [https://arxiv.org/abs/2201.11903], last visited: 2025-12-02.

19. Mitchell, M., "Why AI is Harder Than We Think published April 2021, arXiv preprint, [https://arxiv.org/abs/2104.12871], last visited: 2025-12-02.

20. Wang, X., et al., "Self-Consistency Improves Chain of Thought Reasoning in Language Models published 2023, ICLR 2023, [https://arxiv.org/abs/2203.11171], last visited: 2025-12-02.

21. Schick, T., et al., "Toolformer: Language Models Can Teach Themselves to Use Tools published February 2023, arXiv preprint, [https://arxiv.org/abs/2302.04761], last visited: 2025-12-02.

22. Chen, M., et al., "Evaluating Large Language Models Trained on Code published July 2021, arXiv preprint, [https://arxiv.org/abs/2107.03374], last visited: 2025-12-02.

23. Jiang, E., et al., "Promptmaker: Prompt-based Prototyping with Large Language Models published 2022, CHI 2022, [https://dl.acm.org/doi/10.1145/3491101.3519729], last visited: 2025-12-02.

24. Dakhel, A., et al., "GitHub Copilot AI pair programmer: Asset or Liability? published September 2023, Journal of Systems and Software, vol. 203, [https://www.sciencedirect.com/science/article/pii/S0164121223001292], last visited: 2025-12-02.

25. MDN Web Docs, "CSS Grid Layout Mozilla Developer Network, [https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout], last visited: 2025-12-02.

26. MDN Web Docs, "Manipulating Documents Mozilla Developer Network, [https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Manipulating_documents], last visited: 2025-12-02.

27. Pressman, R., Maxim, B., "Software Engineering: A Practitioner's Approach published 2020, 9th Edition, McGraw-Hill Education.

28. W3C, "Internationalization Best Practices W3C Working Draft, [https://www.w3.org/TR/i18n-html-tech-lang/], last visited: 2025-12-02.

29. Humble, J., Farley, D., "Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation published 2010, Addison-Wesley.

30. Perry, N., et al., "Do Users Write More Insecure Code with AI Assistants? published August 2023, arXiv preprint, [https://arxiv.org/abs/2211.03622], last visited: 2025-12-02.

31. OWASP, "OWASP Top Ten 2021 OWASP Foundation, [https://owasp.org/www-project-top-t last visited: 2025-12-02.

32. Barke, S., et al., "Grounded Copilot: How Programmers Interact with Code-Generating Models published 2023, OOPSLA 2023, [https://arxiv.org/abs/2206.15000], last visited: 2025-12-02.

33. Bender, E., et al., "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? published 2021, FAccT 2021, [https://dl.acm.org/doi/10.1145/3442188.3445922], last visited: 2025-12-02.

34. Ji, Z., et al., "Survey of Hallucination in Natural Language Generation published March 2023, ACM Computing Surveys, vol. 55, no. 12, [https://arxiv.org/abs/2202.03629], last visited: 2025-12-02.

35. MDN Web Docs, "HTML: HyperText Markup Language Mozilla Developer Network, [https://developer.mozilla.org/en-US/docs/Web/HTML], last visited: 2025-12-02.

36. Samuelson, P., "Allocating Ownership Rights in Computer-Generated Works published 1986, University of Pittsburgh Law Review, vol. 47.

37. Search Engine Journal, "AI for SEO: Complete Guide published October 2023, Search Engine Journal, [https://www.searchenginejournal.com/ai-seo/], last visited: 2025-12-02.

38. World Economic Forum, "The Future of Jobs Report 2023 published April 2023, WEF, [https://www.weforum.org/reports/the-future-of-jobs-report-2023], last visited: 2025-12-02.

39. ACM, "Computing Curricula 2020 published December 2020, ACM/IEEE Computer Society, [https://www.acm.org/education/curricula-recommendations], last visited: 2025-12-02.

40. Prather, J., et al., "The Robots Are Here: Navigating the Generative AI Revolution in Computing Education published June 2023, ACM Inroads, vol. 14, no. 2.

41. Forsgren, N., et al., "Accelerate: The Science of Lean Software and DevOps published 2018, IT Revolution Press.

42. Stack Overflow, "2023 Developer Survey published May 2023, Stack Overflow, [https://survey.stackoverflow.co/2023/], last visited: 2025-12-02.

43. Nguyen, N., Nadi, S., "An Empirical Evaluation of GitHub Copilot's Code Suggestions published 2022, MSR 2022, [https://arxiv.org/abs/2205.06537], last visited: 2025-12-02.

44. Ross, S., et al., "Programmer Experiences with GitHub Copilot published August 2023, arXiv preprint, [https://arxiv.org/abs/2303.06104], last visited: 2025-12-02.

45. Google Developers, "Web Fundamentals - Performance Google, [https://developers.google.com/web/fundamentals/performance], last visited: 2025-12-02.

46. Thorp, H., "ChatGPT is fun, but not an author published January 2023, Science, vol. 379, no. 6630.

47. Kalluri, P., "Don't ask if artificial intelligence is good or fair, ask how it shifts power published July 2020, Nature, vol. 583.

48. Buolamwini, J., Gebru, T., "Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification published 2018, FAT 2018.

49. Borji, A., "A Categorical Archive of ChatGPT Failures published February 2023, arXiv preprint, [https://arxiv.org/abs/2302.03494], last visited: 2025-12-02.

50. Pearce, H., et al., "Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions published May 2022, IEEE Symposium on Security and Privacy.

51. Alkaissi, H., McFarlane, S., "Artificial Hallucinations in ChatGPT: Implications in Scientific Writing published February 2023, Cureus.

52. Bass, L., et al., "Software Architecture in Practice published 2021, 4th Edition, Addison-Wesley.

53. Radford, A., et al., "Language Models are Unsupervised Multitask Learners published 2019, OpenAI Blog, [https://openai.com/research/better-language-models], last visited: 2025-12-02.

54. IEEE, "IEEE Code of Ethics IEEE, [https://www.ieee.org/about/corporate/governance/p7-8.html], last visited: 2025-12-02.

55. Brynjolfsson, E., et al., "Generative AI at Work published April 2023, NBER Working Paper, [https://www.nber.org/papers/w31161], last visited: 2025-12-02.

56. Hutson, M., "Could AI help you to write better code? published April 2023, Nature, vol. 616, [https://www.nature.com/articles/d41586-023-00983-1], last visited: 2025-12-02.

Приложения

Приложение А: Кратък речник на термини

- **LLM** - Large Language Model, голям езиков модел
- **Prompt** - Инструкция или въпрос към AI модел
- **Token** - Базова единица текст за обработка от модела
- **Few-shot learning** - Обучение с малко примери
- **Chain-of-thought** - Стъпка-по-стъпка разсъждение
- **Hallucination** - "Измисляне"на факти от AI
- **i18n** - Internationalization, интернационализация
- **OWASP** - Open Web Application Security Project

Приложение Б: Полезни ресурси

- MDN Web Docs: https://developer.mozilla.org/
- W3C Standards: https://www.w3.org/standards/
- OpenAI Documentation: https://platform.openai.com/docs
- OWASP Top 10: https://owasp.org/www-project-top-ten/
- CSS-Tricks: https://css-tricks.com/

Приложение В: Примерни тестови въпроси

1. **Какво представлява "Prompt Engineering"?**
   (а) Процес на хардуерна оптимизация на сървъри
   (б) **Процес на проектиране и оптимизация на инструкции към AI модели**
   (в) Метод за компилиране на JavaScript код
   (г) Техника за управление на бази данни
2. **Коя от следните техники включва предоставяне на примери в промпта?**
   (а) Zero-shot learning
   (б) **Few-shot learning**
   (в) Reinforcement learning
   (г) Unsupervised learning
3. **Какво е "Chain-of-Thought"(CoT) промптване?**

(а) Свързване на няколко AI модела в мрежа

(б) **Насърчаване на модела да обяснява стъпките в разсъжденията си**

(в) Използване на блокчейн технологии в AI

(г) Генериране на безкраен цикъл от промптове

4. **Какъв е основният риск при "Blind AI-Coding"?**

(а) AI моделът ще спре да работи

(б) **Внедряване на код с грешки или уязвимости без проверка**

(в) Кодът ще бъде твърде оптимизиран

(г) Загуба на интернет връзка

5. **Какво представляват "халюцинациите"при LLM?**

(а) Визуални ефекти в интерфейса

(б) **Генериране на невярна или измислена информация, представена като факт**

(в) Прегряване на графичните процесори

(г) Автоматично изтриване на данни

6. **Кой инструмент е подходящ за валидация на AI генериран JavaScript код?**

(а) Photoshop

(б) **ESLint**

(в) Microsoft Word

(г) SQLite

7. **Как AI може да помогне при i18n (интернационализация)?**

(а) Чрез автоматично превеждане на потребителите

(б) **Чрез генериране на JSON структури с преводи за различни езици**

(в) Чрез забрана на достъпа от чужбина

(г) Чрез промяна на часовата зона на сървъра

8. **Какво е "Role Prompting"?**

(а) Задаване на роля на потребителя в системата

(б) **Инструкция към AI модела да действа като експерт в дадена област**

(в) Ролева игра между разработчици

(г) Административен достъп до модела

9. **Защо е важно да се проверяват цитатите, генерирани от AI?**

(а) Защото AI не може да пише на английски

(б) **Защото AI често генерира несъществуващи или грешни източници**

(в) Защото цитатите заемат много място

(г) Няма нужда да се проверяват

10. **Коя е добра практика при използване на AI в CI/CD pipeline?**

(а) Автоматично деплойване на всичко генерирано

(б) **Включване на автоматизирани тестове и security сканиране**

(в) Изключване на всички валидации

(г) Премахване на човешкия контрол

Приложение Г: Изисквания и задание

Тема №187: Prompt engineering и Web разработка с ChatGPT

Цел на проекта: Разработване на интерактивен, семантичен и стилизиран информационен сайт по зададената тема, използвайки съвременни уеб технологии.

Основни изисквания (25 издание):

- Обем: 15 страници (вкл. фигури, код, таблици).
- Език: Български.
- Ресурси: Само на английски език, коректно цитирани.
- Съдържание: Достоверна и проверена информация.
- Технологии: HTML5, CSS3, JavaScript (за уеб версията).
- Структура: Увод, Изложение (с примери), Заключение, Библиография.

Специфични изисквания за темата:

- Дефиниране на Prompt Engineering.
- Анализ на техники (Few-shot, CoT, Role-playing).
- Приложение в уеб разработката (HTML/CSS/JS generation).
- Етични съображения и рискове.
- Бъдеще на професията.